```
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  LLL              TTTTTTTTTTTTTTT     AAAAAAAAA
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  LLL              TTTTTTTTTTTTTTT     AAAAAAAAA
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  LLL              TTTTTTTTTTTTTTT     AAAAAAAAA
DDD         DDD  EEE              LLL                   TTT          AAA       AAA
DDD         DDD  EEE              LLL                   TTT          AAA       AAA
DDD         DDD  EEE              LLL                   TTT          AAA       AAA
DDD         DDD  EEE              LLL                   TTT          AAA       AAA
DDD         DDD  EEE              LLL                   TTT          AAA       AAA
DDD         DDD  EEE              LLL                   TTT          AAA       AAA
DDD         DDD  EEEEEEEEEEE      LLL                   TTT          AAA       AAA
DDD         DDD  EEEEEEEEEEE      LLL                   TTT          AAA       AAA
DDD         DDD  EEEEEEEEEEE      LLL                   TTT          AAA       AAA
DDD         DDD  EEE              LLL                   TTT          AAAAAAAAAAAAAAA
DDD         DDD  EEE              LLL                   TTT          AAAAAAAAAAAAAAA
DDD         DDD  EEE              LLL                   TTT          AAAAAAAAAAAAAAA
DDD         DDD  EEE              LLL                   TTT          AAA       AAA
DDD         DDD  EEE              LLL                   TTT          AAA       AAA
DDD         DDD  EEE              LLL                   TTT          AAA       AAA
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  LLLLLLLLLLLLLLL       TTT          AAA       AAA
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  LLLLLLLLLLLLLLL       TTT          AAA       AAA
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  LLLLLLLLLLLLLLL       TTT          AAA       AAA
```

```
DDDDDDDD    EEEEEEEEEE  LL          TTTTTTTTTT    AAAAAA
DDDDDDDD    EEEEEEEEEE  LL          TTTTTTTTTT    AAAAAA
DD      DD  EE          LL              TT       AA    AA
DD      DD  EE          LL              TT       AA    AA
DD      DD  EE          LL              TT       AA    AA
DD      DD  EE          LL              TT       AA    AA
DD      DD  EEEEEEEE    LL              TT       AA    AA
DD      DD  EEEEEEEE    LL              TT       AA    AA
DD      DD  EE          LL              TT       AAAAAAAAAA
DD      DD  EE          LL              TT       AAAAAAAAAA
DD      DD  EE          LL              TT       AA    AA      ....
DD      DD  EE          LL              TT       AA    AA      ....
DDDDDDDD    EEEEEEEEEE  LLLLLLLLLL      TT       AA    AA      ....
DDDDDDDD    EEEEEEEEEE  LLLLLLLLLL      TT       AA    AA      ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

```
00000001  0000     1 SW_PROCESS=1                          ;
          0000     1         .IF     DF,SW_PROCESS         ;
          0000     2         .TITLE  DELTA- MULTIMODE PROCESS DEBUGGER
          0000     3         .IFF                          ;
          0000     4         .TITLE  XDELTA - EXECUTIVE DEBUGGER
          0000     5         .ENDC                         ;
          0000     6         .IDENT  'V04-000'
          0000     7
          0000     8 ;
          0000     9 ;*****************************************************************************
          0000    10 ;*                                                                           *
          0000    11 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
          0000    12 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
          0000    13 ;*  ALL RIGHTS RESERVED.                                                     *
          0000    14 ;*                                                                           *
          0000    15 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
          0000    16 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
          0000    17 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
          0000    18 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
          0000    19 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
          0000    20 ;*  TRANSFERRED.                                                             *
          0000    21 ;*                                                                           *
          0000    22 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
          0000    23 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
          0000    24 ;*  CORPORATION.                                                             *
          0000    25 ;*                                                                           *
          0000    26 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
          0000    27 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
          0000    28 ;*                                                                           *
          0000    29 ;*                                                                           *
          0000    30 ;*****************************************************************************
          0000    31
          0000    32 ;++
          0000    33 ; FACILITY: EXECUTIVE, DEBUGGING TOOLS
          0000    34 ;
          0000    35 ; ABSTRACT:
          0000    36 ;       THIS MODULE PRODUCES TWO DIFFERENT DEBUGGERS DEPENDING ON THE SETTING
          0000    37 ;       OF THE ASSEMBLY SWITCH, SW_PROCESS.  DELTA IS A MULTIMODE PROCESS
          0000    38 ;       DEBUGGER USING SYSTEM SERVICES WHILE XDELTA IS A STANDALONE EXEC
          0000    39 ;       DEBUGGING TOOL.
          0000    40 ;
          0000    41 ;       COMMAND SYNTAX IS IDENTICAL FOR BOTH VERSIONS EXCEPT FOR ENVIRONMENTAL
          0000    42 ;       DIFFERENCES.  THE SYNTAX IS QUITE TERSE AND SOMEWHAT CRYPTIC AND
          0000    43 ;       IS DOCUMENTED IN THE "GUIDE TO WRITING AN I/O DRIVER".
          0000    44 ;
          0000    45 ; ENVIRONMENT:
          0000    46 ;       DELTA - NORMAL PROCESS ENVIRONMENT, VARIOUS ACCESS MODES.
          0000    47 ;       XDELTA - STANDALONE, RESIDENT, KERNEL MODE, IPL=31
          0000    48 ;       BOTH VERSIONS MUST BE POSITION INDEPENDENT - BEWARE!
          0000    49 ;--
```

```
0000     51          .SBTTL  HISTORY                 ; DETAILED
0000     52 ;
0000     53 ; AUTHOR:       R. HUSTVEDT      CREATION DATE: 15-NOV-76
0000     54 ;
0000     55 ; MODIFIED BY:
0000     56 ;
0000     57 ;       V03-016 WHM0001         Bill Matthews           18-Jul-1984
0000     58 ;               Call CON$GETCHAR and CON$PUTCHAR to do I/O to the console
0000     59 ;               terminal. Call CON$OWNCTY to allocate and CON$REALEASECTY to
0000     60 ;               release the console terminal.
0000     61 ;
0000     62 ;       V03-015 MSH0039         Michael S. Harvey       1-May-1984
0000     63 ;               Adjust image activation SET exception vector index
0000     64 ;               when setting up a DELTA rundown vector so that it
0000     65 ;               won't be lost by a subsequent image activation prior
0000     66 ;               to actual rundown.
0000     67 ;
0000     68 ;       V03-014 MSH0002         Michael S. Harvey       16-Jan-1984
0000     69 ;               Reenable AST delivery in EXIT command to ensure process
0000     70 ;               doesn't hang up when EXIT issued from kernel mode. Also,
0000     71 ;               lengthen input command buffer to match specified maximum
0000     72 ;               length in input QIO.
0000     73 ;
0000     74 ;       V03-013 TCM0003         Trudy C. Matthews       13-Dec-1983
0000     75 ;               Use "Write enable bit" when enabling and disabling
0000     76 ;               console terminal access for venus.
0000     77 ;
0000     78 ;       V03-012 KDM0084         Kathleen D. Morse       27-Sep-1983
0000     79 ;               Add MicroVAX I support to CPUDISP macros.
0000     80 ;
0000     81 ;       V03-011 RLRCPUDISP      Robert L. Rappaport     15-Jun-1983
0000     82 ;               Recode CPUDISP macros to use new format.
0000     83 ;
0000     84 ;       V03-010 MIR1039         Michael I. Rosenblum    27-May-1983
0000     85 ;               Fix non PIC reference in New format QIO
0000     86 ;
0000     87 ;       V03-009 MIR0039         Michael I. Rosenblum    29-Apr-1983
0000     88 ;               Make the process based DELTA use itemlist qio's with
0000     89 ;               The no editing bit set.
0000     90 ;
0000     91 ;       V03-008 JLV0236         Jake VanNoy             25-MAR-1983
0000     92 ;               Make QIO a QIOW in OUTZSTRING so that a read will
0000     93 ;               not block write.
0000     94 ;
0000     95 ;       V03-007 TCM0002         Trudy C. Matthews       16-Feb-1983
0000     96 ;               Correct console enable mask in TCM0001.
0000     97 ;
0000     98 ;       V03-006 ROW0159         Ralph O. Weber          28-JAN-1983
0000     99 ;               Enhance DELTA initialization to set all pages in DELTA to user
0000    100 ;               writable.  This corrects a problem encountered while trying to
0000    101 ;               debug DCL with DELTA.  It also guarantees that DELTA will work
0000    102 ;               in all access modes.  Change limit on rundown handler vector
0000    103 ;               table from 505 to <256-7>.
0000    104 ;
0000    105 ;       V03-005 TCM0001         Trudy C. Matthews       11-Jan-1983
0000    106 ;               Change 11/780 machine check handler to write PR$_SBIFS back
0000    107 ;               to itself to clear error bit.  Add 11/790 machine check
```

```
0000   108 ;                handler; initialize 11/790 console interface registers.
0000   109 ;
0000   110 ;        V03-004 ROW0143         Ralph O. Weber          24-NOV-1982
0000   111 ;                Change process-mode OUTZSTRING to do single QIO for whole
0000   112 ;                string.  Make terminal read/write QIOs do a $WAITEF and retry
0000   113 ;                if insufficient resources error is returned by the QIO system
0000   114 ;                service.  Make reference to CTL$GL_USRUNDWN in SETRUNDWN a
0000   115 ;                weak reference so that DELTA can be linked with SYSINIT.
0000   116 ;                Fix numerous branch destinations broken by the above.  Add
0000   117 ;                call to $IODEF definition macro.
0000   118 ;
0000   119 ;        V03-003 ACG0290         Andrew C. Goldstein,    5-May-1982  20:01
0000   120 ;                Condition rundown handler on user mode startup
0000   121 ;
0000   122 ;        V03-002 ACG0286         Andrew C. Goldstein,    13-Apr-1982  15:12
0000   123 ;                Use privileged rundown handler to reset exception vectors
0000   124 ;
0000   125 ;        V03-001 RIH0097         Richard I. Hustvedt     1-Apr-1982
0000   126 ;                Turn off processor register mode when proceeding.
0000   127 ;
0000   128
```

```
                     0000   130          .SBTTL  DECLARATIONS
                     0000   131
                     0000   132 ;
                     0000   133 ; INCLUDE FILES:
                     0000   134 ;
                     0000   135          $ACBDEF                         ; DEFINE AST CONTROL BLOCK
                     0000   136          $CADEF                          ; DEFINE ASSEMBLY SWITCHES
                     0000   137          $CLIDEF                         ; DEFINE CLI VALUES
                     0000   138          $IODEF                          ; DEFINE I/O FUNCTION CODES
                     0000   139          $IPLDEF                         ; DEFINE IPL VALUES
                     0000   140          $IRPDEF                         ; DEFINE IRP VALUES
                     0000   141          $PCBDEF                         ; DEFINE PROCESS CONTROL BLOCK
                     0000   142          $PRDEF                          ; DEFINE PROCESSOR REGISTERS
                     0000   143          $PRIDEF                         ; DEFINE PRIORITY INCREMENT CLASSES
                     0000   144          $PRTDEF                         ; DEFINE PROTECTION VALUES
                     0000   145          $PSLDEF                         ; DEFINE PSL FIELDS
                     0000   146          $SSDEF                          ; DEFINE SYSTEM SERVICE STATUS CODES
                     0000   147          $TRMDEF                         ; TERMINAL ITEMLIST DEFINITIONS
                     0000   148
                     0000   149 ;
                     0000   150 ; MACROS:
                     0000   151 ;
                     0000   152
                     0000   153 ;
                     0000   154 ; EQUATED SYMBOLS:
                     0000   155 ;
00000008             0000   156 V_F1=8                                  ; FIELD 1 PRESENT FLAG
00000009             0000   157 V_F2=9                                  ; FIELD 2 PRESENT FLAG
0000000A             0000   158 V_F3=10                                 ; FIELD 3 PRESENT FLAG
0000000B             0000   159 V_F4=11                                 ; FIELD 4 PRESENT FLAG
0000000C             0000   160 V_F5=12                                 ; FIELD 5 PRESENT FLAG
0000000D             0000   161 V_INSTR=13                              ; INSTRUCTION DISPLAY MODE
                     0000   162                                        ; (OVERRIDES HEX OR ASCII & CURTYPE)
                     0000   163
00000000             0000   164 V_OPEN=0                                ; OPEN CELL FLAG
00000001             0000   165 V_ASCII=1                               ; ASCII
00000002             0000   166 V_INFIELD=2                             ; FIELD IN PROGRESS
00000003             0000   167 V_TBIT=3                                ; ENABLE TBIT
00000004             0000   168 V_ATBRK=4                               ; AT BREAKPOINT
00000005             0000   169 V_TBITOK=5                              ; TBIT EXPECTED
00000006             0000   170 V_RUB=6                                 ; RUBOUT IN PROGRESS
00000007             0000   171 V_NEGATE=7                              ; NEGATE BIT
0000000F             0000   172 V_PRMODE=15                             ; PROCESSOR REGISTER MODE
0000001F             0000   173 V_PREG=31                               ; PROCESSOR REGISTER FLAG
                     0000   174
00000000             0000   175 RDCR=0                                  ; READ CSR
00000002             0000   176 RDBUF=2                                 ; READ BUFFER
00000004             0000   177 OUTCR=4                                 ; OUTPUT CSR
00000006             0000   178 OUTB=6                                  ; OUTPUT BUFFER
                     0000   179
0000005C             0000   180 BSLSH=92                                ; BACK SLASH CODE
0000000D             0000   181 CR=13                                   ; CARRIAGE RETURN
0000000A             0000   182 LF=10                                   ; LINE FEED
00000027             0000   183 QUOT=39                                 ; QUOTE
0000007F             0000   184 RUBOUT=127                              ; RUBOUT CODE
0000002F             0000   185 SLSH=47                                 ; SLASH CODE
                     0000   186
```

```
                     0000   187
                     0000   188
                     0000   189 ;
                     0000   190 ;       OWN STORAGE:
                     0000   191 ;
                00000000    192         .PSECT  Z$DEBUG_CODE,LONG,PIC,EXE,WRT
                     0000   193
                     0000   194         .IF     DF,SW_PROCESS
00000000  R      0000   195 DELBASE::.LONG    DELBASE-DELBASE             ; RELATIVE PAGE NUMBER OF WRITABLE
00001600'        0004   196         .LONG    <511+DELEND-DELBASE>&^C511; REL PAGE NUMBER OF END OF WRITABLE
00000FC1'        0008   197         .LONG    DELTA_START-DELBASE         ; START ADDRESS
                     000C   198         .ENDC
                     000C   199
                     000C   200 CONTEXT:                                 ;
00000000         000C   201         .LONG    0                           ; BUFFER PADDING
00000060         0010   202 INBUF:  .BLKB    80                          ; INPUT BUFFER
00000000         0060   203 STATUS: .LONG    0                           ; STATUS FLAGS
00000000         0064   204 F1:     .LONG    0                           ; FIELDS
00000000         0068   205 F2:     .LONG    0                           ; 1-5
00000000         006C   206 F3:     .LONG    0                           ;
00000000         0070   207 F4:     .LONG    0                           ;
00000000         0074   208 F5:     .LONG    0                           ;
                     0078   209
00000000         0078   210 MFYFLG: .LONG    0                           ; MODIFY ENABLE FLAG FOR OTHER PROCESS
                     007C   211                                            ; ADDRESS SPACES
00000000         007C   212 PID:    .LONG    0                           ; PID FOR ADDRESS SPACE 0=>SELF
00000000         0080   213 INSLEN: .LONG    0                           ; LENGTH OF PREVIOUS INSTRUCTION
00000000         0084   214 INSBUF: .LONG    0                           ; ADDRESS OF INSTRUCTION STREAM BUFFER
                     0088   215                                            ; (FOR OUTPUT_ADDRESS ROUTINE)
                00   0088   216 FCTR:   .BYTE   0                         ; FIELD COUNTER
                     0089   217
                02   0089   218 DTYPE:  .BYTE   2                         ; DATA TYPE
                02   008A   219 CURTYPE:.BYTE   2                         ; CURRENT TYPE
                     008B   220
                00   008B   221 OPER:   .BYTE   0                         ; OPERATOR
                     008C   222 B:                                         ; BASE OF DATA AREA(CENTER)
00000000         008C   223 CURDOT: .LONG    0                           ; CURRENT LOCATION
00000000         0090   224 QUAN:   .LONG    0                           ; QUANTITY (;Q)
000000A4         0094   225 OUTBUF: .BLKL    4                           ; OUTPUT BUFFER
                     00A4   226 ;
                     00A4   227 ;       REGISTER SAVE AREA
                     00A4   228 ;
                     00A4   229 SAVREG:                                  ; REGISTER SAVE AREA
000000A8         00A4   230         .BLKL    1                           ; R0
000000AC         00A8   231         .BLKL    1                           ; R1
000000B0         00AC   232 SAVR2:  .BLKL    1                           ; R2
000000B4         00B0   233         .BLKL    1                           ; R3
000000B8         00B4   234         .BLKL    1                           ; R4
000000BC         00B8   235         .BLKL    1                           ; R5
000000C0         00BC   236         .BLKL    1                           ; R6
000000C4         00C0   237         .BLKL    1                           ; R7
000000C8         00C4   238         .BLKL    1                           ; R8
000000CC         00C8   239         .BLKL    1                           ; R9
000000D0         00CC   240         .BLKL    1                           ; R10
000000D4         00D0   241         .BLKL    1                           ; R11
000000D8         00D4   242 SAVAP:  .BLKL    1                           ; AP
000000DC         00D8   243         .BLKL    1                           ; (FP)
```

```
000000E0  00DC   244 SAVSP:  .BLKL   1                          ; SP
000000E4  00E0   245 SAVPC:  .BLKL   1                          ; PC
000000E8  00E4   246 SAVPSL: .BLKL   1                          ; PSL
000000EA  00E8   247 SAVOCR: .BLKW   1                          ; OUTPUT CSR SAVE
000000EC  00EA   248 SAVRCR: .BLKW   1                          ; INPUT CSR SAVE
          00EC   249 ASTEN:                                     ; AST ENABLE SAVE LOCATION
000000F0  00EC   250 SAVRXCS:.BLKL   1                          ; CONSOLE RECEIVER STATUS
          00F0   251
000000E4  00F0   252 CONTEXTSZ=.-CONTEXT                        ; SIZE OF PER MODE CONTEXT AREA
          00F0   253 ;
          00F0   254 ;       RESERVE SPACE FOR MULTIPLE MODE CONTEXT AREA
          00F0   255 ;
          00F0   256         .IF     DF,SW_PROCESS             ;
          00F0   257         .REPT   3                         ;
          00F0   258         .BLKB   CONTEXTSZ                 ; FOR EXEC,SUPER AND USER
          00F0   259 SAV...= .
          00F0   260         .=.-CONTEXTSZ+<DTYPE-CONTEXT>     ; POINT AT DTYPE,CURTYP
          00F0   261         .BYTE   2,2                       ; SET TYPE TO LONGWORD
          00F0   262         .=SAV...                          ; RESTORE LOCATION COUNTER
000001D4  00F0   263         .ENDR                             ;
          039C   264         .ENDC                             ;
          039C   265
          039C   266
          039C   267 ;
          039C   268 ;       BREAK POINT DATA
          039C   269 ;
          039C   270
      00  039C   271 OVROPC: .BYTE   0                          ; OPCODE IN STEP-OVER BREAKPOINT
          039D   272         .ALIGN  LONG
          03A0   273
0000039C  03A0   274 BRKADR=.-4
          03A0   275         .IF     NDF,SW_PROCESS
          03A0   276 XDELIBRK::
          03A0   277         .LONG   INI$BRK                   ; ADDRESS OF INITIAL BREAKPOINT
          03A0   278         .IFF                              ; FOR PROCESS VERSION
00000000  03A0   279 INIBRKA:.LONG   0                          ; INITIAL BREAKPOINT
          03A4   280         .ENDC                             ;
000003C0  03A4   281         .BLKL   7                         ; OTHER BREAK POINT ADDRESSES
00000008  03C0   282 NBRK=<.-4-BRKADR>/4                        ; NUMBER OF BREAKPOINTS
000003C4  03C0   283 OVRADR: .BLKL   1                          ; TEMPORARY BREAKPOINT FOR STEP-OVER
00000001  03C4   284 NTMPBRK=1                                  ; NUMBER OF TEMPORARY BREAKPOINTS
000003C3  03C4   285 BRKOP=.-1                                  ; SAVED OPCODE
      01  03C4   286         NOP                               ; INITIAL OPCODE
000003CC  03C5   287         .BLKB   7                         ; REMAINING OPCODES
000003CD  03CC   288         .BLKB   1                         ; TEMPORARY BREAKPOINT OPCODE
          03CD   289
000003C9  03CD   290 BRKDSP=.-4
000003ED  03CD   291         .BLKL   8                         ; DISPLAY LOCATION START
000003E9  03ED   292 BRKCOM=.-4
0000040D  03ED   293         .BLKL   8                         ; COMMAND START
          040D   294
00000419  040D   295 XREGV:  .BLKL   3                          ; X REGISTER VECTOR
          0419   296         .IF     NDF,SW_PROCESS            ;
          0419   297 XDEL_LOADBASE::                          ; BASE OF LOADABLE CPU DEPENDANT CODE
          0419   298         .LONG   0                         ; X3 = BASE OF SYSLOA CODE
          0419   299         .LONG   SCH$GL_CURPCB             ; X4 = CURRENT PCB ADDRESS
          0419   300         .LONG   SCH$GL_PCBVEC             ; X5 = BASE OF PCB VECTOR
```

```
                     0419   301          .LONG   PFN$AW_SWPVBN           ; X6 = SWAP VBN
                     0419   302          .LONG   PFN$AL_PTE              ; X7 = PTE BACK POINTER
                     0419   303          .LONG   PFN$AL_BAK              ; X8 = BACKUP ADDRESS
                     0419   304          .LONG   PFN$AW_REFCNT           ; X9 = REFERENCE COUNT
                     0419   305          .LONG   PFN$Ax_FLINK            ; XA = FORWARD LINK
                     0419   306          .LONG   PFN$Ax_BLINK            ; XB = BACK LINK
                     0419   307          .LONG   PFN$AB_STATE            ; XC = STATE
                     0419   308          .LONG   PFN$AB_TYPE             ; XD = TYPE
                     0419   309 XDS$GL_XESTRING::
                     0419   310          .LONG   XDS$GT_WORD_PFN         ; XE;E WITH X0 = PFN , DEFAULT TO WORD ARRAY
                     0419   311 XDS$GL_XFSTRING::
                     0419   312          .LONG   XDS$GT_WORD_PFN         ; XF;E WITH R0 = PFN , DEFAULT TO WORD ARRAY
                     0419   313 MCHKSAV:.BLKL   1                       ; SAVED CONTENT OF MACHINE CHECK VECTOR
                     0419   314          .IFF                            ; FOR PROCESS VERSION
0000044D             0419   315          .BLKL   13                      ;
00000455             044D   316 TTIOSB:  .BLKL   2                       ; IO STATUS BLOCK FOR TERMINAL READ
00000459             0455   317 TTCHAN:  .BLKL   1                       ; CHANNEL NUMBER
00000002             0459   318 TTNAMD:  .LONG   2                       ; DESCRIPTOR OF INPUT/OUTPUT DEVICE
00000461             045D   319          .BLKL   1 ;TTNAMD+8             ; (ADDRESS SET BY INITIALIZATION)
      54 54          0461   320          .ASCII  /TT/                    ;
                     0463   321 TTITMLST:                                ; THE ITEMLIST TO ALLOW DELTA TO TURN OFF ED
         0000        0463   322          .WORD   0
         0000        0465   323          .WORD   TRM$_MODIFIERS          ; SPECIFY THE MODIFIERS
00008000             0467   324          .LONG   TRM$M_TM_NOEDIT         ; SPECIFY NO EDITING
00000000             046B   325          .LONG   0
           0010'     046F   326          .WORD   TERMASKLEN              ; LENGTH OF TERMINATOR MASK
           0003      0471   327          .WORD   TRM$_TERM               ;SPECIFY THE TERMINATOR MASK
                     0473   328 TERMASKADR:                              ; ALLOW FOR RELOCATION
0000047B             0473   329          .BLKL   2
00000018             047B   330 TTITMLSTLEN=.-TTITMLST
                     047B   331 DBGINPUT:
00000009             047B   332          .LONG   9                       ; DESCRIPTOR OF DEFAULT INPUT/OUTPUT
00000483             047F   333          .BLKL   1 ;DBGINPUT+8
41 54 4C 45 44 24 47 42 44   0483   334          .ASCII  /DBG$DELTA/             ; FIRST DEFAULT DELTA INPUT
                     048C   335 TRNINPUT:
00000040             048C   336          .LONG   64                      ; TRANSLATED DBG$DELTA
00000494             0490   337          .BLKL   1 ;TRNINPUT+8           ; (ADDRESS SET BY INITIALIZATION)
000004D4             0494   338          .BLKB   64
                     04D4   339 DBGACTIVE:                              ; ACTIVE FLAGS BY ACCESS MODE
00000000             04D4   340          .LONG   0                       ;
                     04D8   341 EXITBLK:                                ; EXIT HANDLER BLOCK
00000000             04D8   342          .LONG   0                       ;
000004E0             04DC   343 EXIHADR:.BLKL   1 ;EXIHANDLE            ; EXIT HANDLER (ADDRESS SET BY INIT)
00000001             04E0   344          .LONG   1                       ; ARGUMENT COUNT
000004E8             04E4   345 EXCODA: .BLKL   1 ;EXITCODE             ; ADDRESS TO STORE STATUS (ADDRESS SET BY IN
                     04E8   346 EXITCODE:                               ;
00000001             04E8   347          .LONG   1                       ; RECEIVER FOR EXIT CODE
                     04EC   348 KCOND_PRIMARY:
00000000             04EC   349          .LONG   0                       ; PREVIOUS KERNEL PRIMARY HANDLER
                     04F0   350 ECOND_PRIMARY:
00000000             04F0   351          .LONG   0                       ; PREVIOUS EXEC PRIMARY HANDLER
                     04F4   352 SCOND_PRIMARY:
00000000             04F4   353          .LONG   0                       ; PREVIOUS SUPER PRIMARY HANDLER
                     04F8   354 KCOND_LASTCHANC:
00000000             04F8   355          .LONG   0                       ; PREVIOUS KERNEL LAST CHANCE HANDLER
                     04FC   356 ECOND_LASTCHANC:
00000000             04FC   357          .LONG   0                       ; PREVIOUS EXEC LAST CHANCE HANDLER
```

```
                 0500   358 SCOND_LASTCHANC:
00000000         0500   359         .LONG   0                               ; PREVIOUS SUPER LAST CHANCE HANDLER
                 0504   360 TERMASK:                                        ; TERMINATOR MASK DESCRIPTOR
08002600         0504   361         .LONG   <1a9>!<1a10>!<1a13>!<1a27>      ; TAB,LF,CR,ESC
20008002         0508   362         .LONG   <1a1>!<1a2>!<1a15>!<1a29>       ; '!', '%', '/', '='
00088000         050C   363         .LONG   <1a15>!<1a19>                   ; '0', '5'
00000000         0510   364         .LONG   0                               ;
00000010         0514   365 TERMASKLEN=.-TERMASK
                 0514   366         .ENDC                                   ;
                 0514   367 ;
                 0514   368 ; LIST OF OPCODES WHICH CALL ROUTINES
                 0514   369 ;
                 0514   370 OVEROPCODES:
      10         0514   371         .BYTE   ^X10                            ; BSBB
      16         0515   372         .BYTE   ^X16                            ; JSB
      30         0516   373         .BYTE   ^X30                            ; BSBW
      FA         0517   374         .BYTE   ^XFA                            ; CALLG
      FB         0518   375         .BYTE   ^XFB                            ; CALLS
00000005         0519   376 OVEROPCLEN = .-OVEROPCODES
```

```
                                0519   378          .SBTTL  PRIMARY COMMAND CHARACTER SWITCH
                                0519   379
                                0519   380 ;
                                0519   381 ;       PRIMARY CHARACTER LIST
                                0519   382 ;
                                0519   383 PRIMARY:
42 41 39 38 37 36 35 34 33 32 31 30  0519   384          .ASCII  /0123456789ABCDEF/      ; DECIMAL AND HEX CHARS
                   46 45 44 43  0525
                             2E  0529   385          .ASCII  /./                     ; DOT - CURRENT LOCATION
                             2C  052A   386          .ASCII  /,/                     ; COMMA - FIELD SEPARATOR
                       00000012  052B   387 OPERBAS=.-PRIMARY                       ; OPERATORS
                             2B  052B   388          .ASCII  /+/                     ; PLUS - ADD
                             20  052C   389          .ASCII  / /                     ; BLANK - SAME AS PLUS
                             40  052D   390          .ASCII  /@/                     ; SHIFT OPERATOR
                             2A  052E   391          .ASCII  /*/                     ; MULTIPLY OPERATOR
                             25  052F   392          .ASCII  /%/                     ; DIVIDE OPERATOR
                             2D  0530   393          .ASCII  /-/                     ; MINUS - SUBTRACT OPERATOR
                             5B  0531   394          .ASCII  /[/                     ; LBRACKET - LEFT BRACKET
                                 0532   395 TERM:                                   ; BASE OF TERMINATOR LIST
                             09  0532   396          .ASCII  <9>                     ; TAB - INDIRECT
                             0A  0533   397          .ASCII  <10>                    ; LINEFEED -
                             0D  0534   398          .ASCII  <CR>                    ; RETURN -
                             2F  0535   399          .ASCII  '/'                     ; SLASH - OPEN FOR DISPLAY
                             22  0536   400          .ASCII  '"'                     ; DOUBLE QUOTE - OPEN FOR ASCII DISPLAY
                             3D  0537   401          .ASCII  /=/                     ; EQUALS - DISPLAY
                             1B  0538   402          .ASCII  <27>                    ; ESCAPE - PREVIOUS LOCATION
                             53  0539   403          .ASCII  /S/                     ; STEP
                             4F  053A   404          .ASCII  /O/                     ; STEP-OVER ROUTINE
                             21  053B   405          .ASCII  /!/                     ; DISPLAY INSTRUCTION
                       0000000A  053C   406 NTERM=.-TERM                           ; NUMBER OF TERMINATORS
                             3B  053C   407          .ASCII  <59>                    ; SEMI - INITIATE SECONDARY
                             3A  053D   408          .ASCII  /:/                     ; COLON - SEPARATE PID FORM ADDRESS
                             50  053E   409          .ASCII  /P/                     ; P - PROCESSOR REGISTER PREFIX
                             51  053F   410          .ASCII  /Q/                     ; Q - LAST QUANTITY
                             27  0540   411          .ASCII  /'/                     ; QUOTE - BEGIN CHAR STRING
                             52  0541   412          .ASCII  /R/                     ; REGISTER PREFIX
                             47  0542   413          .ASCII  /G/                     ; G - GLOBAL PREFIX
                             48  0543   414          .ASCII  /H/                     ; H - HIGH, P1 SPACE PREFIX
                             58  0544   415          .ASCII  /X/                     ; X REGISTER PREFIX
                       0000002C  0545   416 NPRIM=.-PRIMARY                        ; NUMBER OF PRIMARY COMMANDS
                                 0545   417
```

```
                        0545   419          .SBTTL  PRIMARY COMMAND SCANNER
                        0545   420
                        0545   421 ;
                        0545   422 ;       PRIMARY COMMAND SCANNER
                        0545   423 ;
                        0545   424
                        0545   425
00 0D 0A 3F 48 45 0D 0A 0545   426 OUTER:  .ASCIZ  <LF><CR>/EH?/<LF><CR>
                        054D   427
                   0000 054D   428 DCOM:   .WORD                           ; CALL ENTRY POINT
                        054F   429          .IF     DF,SW_PROCESS          ; FOR PROCESS VERSION ONLY
        6D  1241'CF  9E 054F   430          MOVAB   W^DBGEXCEP,(FP)        ; SET CONDITION HANDLER ADDRESS
                        0554   431          .ENDC                          ;
              13  11    0554   432          BRB     SCANP                  ; ENTER SCANP
        54  EC AF  9E   0556   433 ERROR:   MOVAB   OUTER,R4               ; SET ADDR OF CONTROL STRING
            0388  30    055A   434          BSBW    OUTZSTRING             ; OUTPUT ASCIZ STRING
     5E     5D  D0     055D   435 SUPERST: MOVL    SP,SP                  ; RESET STACK
     59  84 AB  9E     0560   436          MOVAB   INBUF-B(R11),R9        ; RESET STRING ADDRESS
            69  94     0564   437          CLRB    (R9)                   ; AND FORCE READ
            0563  30    0566   438          BSBW    RESET                  ; RESET SCANNER
              02  10    0569   439 SCANP:   BSBB    NEXTP                  ; SCAN INPUT
              FC  11    056B   440          BRB     SCANP                  ; SCAN IT ALL
                        056D   441 NEXTP:                                 ; PROCESS NEXT PRIMARY CHAR
            042B  30    056D   442          BSBW    GETCHAR                ; GET CHARACTER
A4  AF      2C  58  3A  0570   443          LOCC    R8,#NPRIM,PRIMARY      ; CHECK IT
              DF  13    0575   444          BEQL    ERROR                  ; NOT FOUND, ERROR
     50     2C  50  C3  0577   445          SUBL3   R0,#NPRIM,R0           ; RATIONALIZE INDEX
                        057B   446          CASE    R0,LIMIT=#16,<-        ;
                        057B   447                  DOT,-                  ; DOT - CURRENT LOCATION
                        057B   448                  COMMA,-                ; COMMA - FIELD SEPARATOR
                        057B   449                  OPERATOR,-             ; PLUS - ADD OPERATOR
                        057B   450                  OPERATOR,-             ; BLANK - ADD OPERATOR
                        057B   451                  OPERATOR,-             ; @ - SHIFT OPERATOR
                        057B   452                  OPERATOR,-             ; * - MULTIPLY OPERATOR
                        057B   453                  OPERATOR,-             ; % - DIVIDE OPERATOR
                        057B   454                  NEGATE,-               ; MINUS - SUBTRACT/NEGATE
                        057B   455                  LBRACKET,-             ; LEFT BRACKET - MODE SELECT
                        057B   456                  TAB,-                  ; TAB - INDIRECT
                        057B   457                  LINEFEED,-             ; LINE FEED - NEXT LOCATION
                        057B   458                  RETURN,-               ; RETURN - CLOSE OPEN CELL
                        057B   459                  SLASH,-                ; SLASH - OPEN FOR DISPLAY
                        057B   460                  DQUOTE,-               ; DOUBLE QUOTE - OPEN FOR ASCII DISPLAY
                        057B   461                  EQUALS,-               ; EQUALS - DISPLAY VALUE
                        057B   462                  ESCAP,-                ; ESCAPE - PREVIOUS LOCATION
                        057B   463                  STEP,-                 ; 'S' - SINGLE STEP
                        057B   464                  STEPOVER,-             ; 'O' - STEP OVER ROUTINE CALL
                        057B   465                  INSTR,-                ; '!' - DISPLAY INSTRUCTION
                        057B   466                  SEMI,-                 ; SEMI COLON - SECONDARY COMMAND
                        057B   467                  COLON,-                ; COLON - SEPARATE PID FROM ADDRESS
                        057B   468                  PREG,-                 ; 'P' - PROCESSOR REGISTER
                        057B   469                  QUANT,-                ; 'Q' - LAST QUANTITY
                        057B   470                  QUOTE,-                ; QUOTE - BEGIN ASCII STRING
                        057B   471                  REGISTER,-             ;
                        057B   472                  GLOBL,-                ; G - GLOBAL PREFIX
                        057B   473                  HIGH,-                 ; H - PT SPACE PREFIX
                        057B   474                  XREG,-                 ; X REGISTER
                        057B   475                  >                      ;
```

```
                10   50   B1   05B7   476          CMPW    R0,#16                  ; IS NUMBER > RADIX
                     9A   18   05BA   477          BGEQ    ERROR                   ; YES
                56   10   C4   05BC   478          MULL    #16,R6                  ; SCALE BY RADIX
                56   50   C0   05BF   479          ADDL    R0,R6                   ; AND ADD NEW DIGIT
                6A   04   C8   05C2   480 INFLD:    BISL    #<1@V_INFIELD>,(R10)    ; NOTE FIELD INPUT
                     05   05C5   481          RSB                             ; NEXT PRIMARY CHARACTER
                          05C6   482
                          05C6   483
          54    01   1F   9C   05C6   484 GLOBL:    ROTL    #31,#1,R4               ; GENERATE SYSTEM SPACE PREFIX
                     07   11   05CA   485          BRB     PRE1                    ; MERGE WITH COMMON
    54    7FFE0000   8F   D0   05CC   486 HIGH:     MOVL    #^X7FFE0000,R4          ; P1 SPACE BASE ADDRESS
                     06   10   05D3   487 PRE1:     BSBB    ENDEXPR                 ; END EXPRESSION
                56   54   D0   05D5   488          MOVL    R4,R6                   ; SET INTO ACCUM
                  E7 AF   9F   05D8   489          PUSHAB  INFLD                   ; RETURN THROUGH INFLD
                          05DB   490 ;        BRB     ENDEXPR
                          05DB   491
```

```
                        05DB   493          .SBTTL  ENDEXPR - END EXPRESSION
                        05DB   494
                        05DB   495 ;
                        05DB   496 ;
                        05DB   497 ;
                        05DB   498 ENDEXPR:
03 6A   07   E5         05DB   499          BBCC    #V_NEGATE,(R10),5$      ;SKIP IF NOT NEGATE
   56   56   CE         05DF   500          MNEGL   R6,R6                   ;NEGATE ACCUMULATOR
        06   10         05E2   501 5$:      BSBB    10$                     ;PERFORM OPERATION
        56   D4         05E4   502          CLRL    R6                      ;CLEAR ACCUMULATOR
     FF AB   94         05E6   503          CLRB    OPER-B(R11)             ;INIT OPERATOR
             05         05E9   504          RSB                             ;AND RETURN
                        05EA   505 10$:     CASE    OPER-B(R11),TYPE=B,<-   ;DO OPERATION
                        05EA   506                  ADD,-                   ;ADD, PLUS
                        05EA   507                  ADD,-                   ;BLANK, PLUS
                        05EA   508                  SHFT,-                  ;SHIFT, @
                        05EA   509                  MUL,-                   ;MULTIPLY, *
                        05EA   510                  DIV,-                   ;DIVIDE, /
                        05EA   511                  >
57  57   56   78         05F9   512 SHFT:    ASHL    R6,R7,R7                ;SHIFT
             05         05FD   513          RSB                             ;AND EXIT
     57   56   C4         05FE   514 MUL:     MULL    R6,R7   ; MULTIPLY
             05         0601   515          RSB                             ;AND EXIT
     57   56   C6         0602   516 DIV:     DIVL    R6,R7   ; DIVIDE
             05         0605   517          RSB                             ;AND EXIT
     57   56   C0         0606   518 ADD:     ADDL    R6,R7   ; ADD
             05         0609   519          RSB                             ;AND EXIT
                        060A   520
```

```
                        060A   522          .SBTTL  SLASH - OPEN CELL
                        060A   523
                        060A   524 ;
                        060A   525 ;       OPEN SPECIFIED CELL
                        060A   526 ;
                        060A   527 DQUOTE:                                      ; DISPLAY IN ASCII
            6A   02   88 060A   528          BISB    #<1@V_ASCII>,(R10)         ; SET ASCII FLAG
                 05   11 060D   529          BRB     OPEN                       ;
                        060F   530
                        060F   531 SLASH:                                       ;
        6A  2002 8F   AA 060F   532          BICW    #<1@V_ASCII>!<1@V_INSTR>,(R10)  ; CLEAR ASCII DISPLAY MODE
        6A  2000 8F   AA 0614   533 OPEN:    BICW    #<1@V_INSTR>,(R10)         ; CLEAR INSTRUCTION FLAG
                 4E   10 0619   534          BSBB    ENDFIELD                   ; TERMINATE FIELD
          06 6A      08 E0 061B   535          BBS     #V_F1,(R10),5$             ; ADDR SPECIFIED?
          6B   04   AB D0 061F   536          MOVL    QUAN-B(R11),CURDOT-B(R11)  ;       NO, GO INDIRECT
                 04   11 0623   537          BRB     10$                        ; AND DISPLAY CONTENT
          6B   D8   AB D0 0625   538 5$:      MOVL    F1-B(R11),CURDOT-B(R11)    ; SET NEW DOT
    50    6A   01    0F EF 0629   539 10$:     EXTZV   #V_PRMODE,#1,(R10),R0      ; GET PROCESSOR REGISTER MODE FLAG
    6A    01   1F    50 F0 062E   540          INSV    R0,#V_PREG,#1,(R10)        ; AND MOVE TO SEMI-PERMANENT COPY
              00B7   30 0633   541          BSBW    LOCOUT                     ; OUTPUT AND OPEN
          22 6A      09 E1 0636   542          BBC     #V_F2,(R10),RSET           ; RANGE SPECIFIED?
          6B   DC   AB D1 063A   543 15$:     CMPL    F2-B(R11),CURDOT-B(R11)    ; CHECK FOR END
                 1C   15 063E   544          BLEQ    RSET                       ; YES
               00A5   30 0640   545          BSBW    NEXTLOC                    ; INCREMENT TO NEXT DOT
                 F5   11 0643   546          BRB     15$                        ; AND CONTINUE
               FF0E   31 0645   547 ERR4:    BRW     ERROR                      ; DECLARE ERROR
                        0648   548
```

```
                     0648   550          .SBTTL  RETURN - CLOSE CURRENT OPEN CELL
                     0648   551
                     0648   552 ;
                     0648   553 ;       RETURN - CLOSE CURRENT OPEN CELL
                     0648   554 ;
                     0648   555
                     0648   556 RETURN:                                      ;
           1F   10   0648   557          BSBB    ENDFIELD                   ; TERMINATE CURRENT FIELD
                     064A   558          .ENABL  LSB                        ;
     11 6A 00   E5   064A   559          BBCC    #V_OPEN,(R10),10$          ; SKIP IF NONE OPEN
6A 2002 8F   B3   064E   560          BITW    #<T@V_ASCII>!<1@V_INSTR>,(R10)  ; IF ASCII OR INSTRUCTION
        07   12   0653   561          BNEQ    RSET                       ; DISPLAY MODE SKIP STORE OPERATION
  03 6A 08   E1   0655   562          BBC     #V_F1,(R10),RSET           ; SKIP IF NOTHING TO STORE
      0887   30   0659   563          BSBW    DEPOSIT                    ; DEPOSIT
      046D   31   065C   564 RSET:    BRW     RESET                      ; RESET SCANNER
F9 6A 08   E1   065F   565 10$:     BBC     #V_F1,(R10),RSET           ; DONE IF NO INPUT
      045E   31   0663   566          BRW     EQL1                       ; OTHERWISE OUTPUT
                     0666   567          .DSABL  LSB                        ;
```

```
                     0666   569          .SBTTL  ENDFIELD - TERMINATE CURRENT FIELD
                     0666   570
                     0666   571 ;
                     0666   572 ;       COMMA TERMINATE CURRENT FIELD
                     0666   573 ;
          FF59   30  0666   574 COMMA:   BSBW    INFLD                   ; ZERO IF NULL FIELD
                     0669   575
                     0669   576 ;
                     0669   577 ;       TERMINATE CURRENT FIELD
                     0669   578 ;
                     0669   579 ENDFIELD:                                ;
    16 6A    02   E5 0669   580          BBCC    #V_INFIELD,(R10),10$    ; CLEAR PENDING FIELD
        FF68   30  066D   581          BSBW    ENDEXPR                 ; END EXPRESSION
  50    FC AB   9A 0670   582          MOVZBL  FCTR-B(R11),R0          ; GET FIELD POINTER
CC 01 AA    50   E2 0674   583          BBSS    R0,1(R10),ERR4          ; ERROR IF TOO MANY FIELDS
 DB AB40    57   D0 0679   584          MOVL    R7,F1-B(R11)[R0]        ; STORE FIELD VALUE
       FC AB   96 067E   585          INCB    FCTR-B(R11)             ; INCREMENT FIELD COUNTER
          56   7C 0681   586          CLRQ    R6                      ; CLEAR ACCUMULATORS
               05 0683   587 10$:     RSB                             ; RETURN
                  0684   588
```

```
                     0684    590          .SBTTL  FETCH - OBTAIN DATA SPECIFIED
                     0684    591
                     0684    592 ;
                     0684    593 ;       FETCH SPECIFIED DATA
                     0684    594 ;
                     0684    595 FETCH:
                     0684    596          BBS     #V_PREG,(R10),40$       ; BR IF PROCESSOR REGISTER
                     0688    597          .IF     DF,SW_PROCESS
                     0688    598          TSTL    PID-B(R11)              ; CHECK FOR PROCESS GET
                     068B    599          BNEQ    50$                     ; BR IF YES
                     068D    600          .ENDC
                     068D    601          CASE    CURTYPE-B(R11),TYPE=B,<-        ; OPERATE ON TYPE
                     068D    602                  10$,-                   ; BYTE
                     068D    603                  20$,-                   ; WORD
                     068D    604                  30$,-                   ; LONG
                     068D    605                  >
                     0698    606 10$:     MOVZBL  @CURDOT-B(R11),QUAN-B(R11)      ; GET BYTE
                     069D    607          RSB                             ; RETURN
                     069E    608 20$:     MOVZWL  @CURDOT-B(R11),QUAN-B(R11)      ; GET WORD
                     06A3    609          RSB                             ; RETURN
                     06A4    610 30$:     MOVL    @CURDOT-B(R11),QUAN-B(R11)      ; GET LONGWORD
                     06A9    611          RSB                             ; RETURN
                     06AA    612          .IF     NDF,SW_PROCESS
                     06AA    613 40$:     MFPR    CURDOT-B(R11),QUAN-B(R11)       ; GET PROCESSOR REGISTER
                     06AA    614          RSB
                     06AA    615          .IFF                            ; FALSE IF PROCESS VERSION
                     06AA    616 40$:
                     06AA    617          $CMKRNL_S       B^FTCHPREG,(AP) ; CALL IN KERNEL MODE TO FETCH
                     06B6    618          RSB
                     06B7    619 50$:     BRW     FETCHP                  ; FETCH FROM FOREIGN PROCESS
                     06BA    620          .ENDC
                     06BA    621
                     06BA    622          .IF     DF,SW_PROCESS           ;
                     06BA    623 FTCHPREG:                                ;
                0000 06BA    624          .WORD   0                       ; ENTRY MASK
6D   0F81'CF   9E    06BC    625          MOVAB   W^PREXC,(FP)            ; SET EXCEPTION HANDLER
  04 AB    68    DB  06C1    626          MFPR    CURDOT-B(R11),QUAN-B(R11)       ; GET PROCESSOR REGISTER
     50    01    D0  06C5    627          MOVL    #1,R0                   ; RETURN SUCCESS
                 04  06C8    628          RET                             ;
                     06C9    629
                     06C9    630          .ENDC                           ;
```

```
                        06C9   632          .SBTTL  NEXTDOT - INCREMENT CURRENT LOCATION
                        06C9   633
                        06C9   634 ;
                        06C9   635 ;       INCREMENT TO NEXT LOCATION
                        06C9   636 ;
                        06C9   637 NEXTDOT:                                             ;
      10 6A    0D   E0  06C9   638          BBS     #V_INSTR,(R10),20$                 ; BRANCH IF INSTRUCTION MODE
         51    01   D0  06CD   639          MOVL    #1,R1                              ; ASSUME UNIT INCREMENT
               6A   D5  06D0   640          TSTL    (R10)                              ; CHECK FOR PREG
               05   19  06D2   641          BLSS    10$                                ; YES, USE UNIT INCREMENT
   51  51     FE   AB  9C  06D4   642          ROTL    CURTYPE-B(R11),R1,R1               ; FORM INCREMENT
         6B    51   C0  06D9   643 10$:      ADDL    R1,CURDOT-B(R11)                   ; AND ADD TO DOT
               05   06DC   644          RSB                                        ; RETURN
         6B    F4   AB  C0  06DD   645 20$:      ADDL    INSLEN-B(R11),CURDOT-B(R11) ; SKIP OVER PREVIOUS INSTRUCTION
               05   06E1   646          RSB
                        06E2   647
```

```
                        06E2    649          .SBTTL  OUTPUT - DISPLAY CONTENT
                        06E2    650 ;
                        06E2    651 ;       OUTPUT CONTENT
                        06E2    652 ;
                        06E2    653 OUTBB:
                1C 0C 04 06E2    654          .BYTE   4,12,28                 ; STARTING DIGIT LIST
                        06E5    655
                        06E5    656          .SBTTL  LINE FEED - DISPLAY NEXT
                        06E5    657
                        06E5    658 LINEFEED:                                ;
              FF60   30 06E5    659          BSBW    RETURN                  ; CLOSE OPEN CELL
                        06E8    660 NEXTLOC:                                 ; PROMPT WITH NEXT LOCATION
                DF   10 06E8    661          BSBB    NEXTDOT                 ; INCREMENT LOCATION
                        06EA    662 LOCPROMPT:                               ; DISPLAY ADDR/CONTENT
              0161   30 06EA    663          BSBW    OUTPUTA                 ; OUTPUT ADDRESS
          31 6A   0D   E0 06ED    664 LOCOUT: BBS     #V_INSTR,(R10),OUTINS   ; BRANCH IF INSTRUCTION MODE
                91   10 06F1    665          BSBB    FETCH                   ; FETCH CONTENT
             6A   01   88 06F3    666          BISB    #<1@V_OPEN>,(R10)       ; INDICATE OPEN CELL
                        06F6    667
                        06F6    668 OUTPUT:                                  ;
           51   FE AB   9A 06F6    669          MOVZBL  CURTYPE-B(R11),R1       ; GET TYPE
        52   E4 AF41   9A 06FA    670          MOVZBL  OUTBB[R1],R2            ; INIT DIGIT SELECTOR
          53   04 AB   D0 06FF    671          MOVL    QUAN-B(R11),R3          ; GET QUANTITY TO DISPLAY
          05 6A   01   E0 0703    672          BBS     #V_ASCII,(R10),10$      ; CHECK FOR ASCII OUT
              01C1   30 0707    673          BSBW    OUTCOM                  ; OUTPUT NUMBER IN HEX
                0F   11 070A    674          BRB     20$                     ; AND EXIT THROUGH OUTSPACE
          08 AB   53   D0 070C    675 10$:     MOVL    R3,OUTBUF-B(R11)        ; PUT STRING IN BUFFER
        52   01   51   78 0710    676          ASHL    R1,#1,R2                ; GET COUNT
             08 AB42   94 0714    677          CLRB    OUTBUF-B(R11)[R2]       ; MARK END OF STRING
              01C6   30 0718    678          BSBW    OUTZBUF                 ; OUTBUT ASCIIZ BUFFER
              026E   31 071B    679 20$:     BRW     OUTSPACE                ; FOLLOW WITH SPACE
                        071E    680
```

```
                                071E   682          .SBTTL  OUTINS - OUTPUT INSTRUCTION
                                071E   683 ;
                                071E   684 ;       OUTPUT RANGE OF INSTRUCTIONS
                                071E   685 ;
                00 09 20 20     071E   686 SPACES: .ASCIZ  '       '                ; 2 SPACES AND A TAB
                                0722   687
                                0722   688          .WEAK   LIB$INS_DECODE          ; INSTRUCTION DECODE IS OPTIONAL
                                0722   689
   50   00000000'GF   9E        0722   690 OUTINS: MOVAB   G^LIB$INS_DECODE,R0     ; GET ADDRESS OF INSTRUCTION DECODER
                  07   12        0729   691          BNEQ    5$                     ; BRANCH IF LINKED WITH DECODER
      6A    2000 8F   AA        072B   692          BICW    #1@V_INSTR,(R10)       ; SUPPRESS INSTRUCTION MODE
                  BB   11        0730   693          BRB     LOCOUT                 ; AND PRINT 1ST LONGWORD OF INS STREAM
   5E   00000052 8F   C2        0732   694 5$:       SUBL    #32+50,SP              ; ALLOCATE SPACE FOR INSTRUCTION STREAM
                                0739   695                                         ; AND DECODE OUTPUT BUFFER
         F8 AB    5E   D0        0739   696          MOVL    SP,INSBUF-B(R11)       ; SAVE ADDRESS FOR OUTPUT_ADDRESS
            54    08   D0        073D   697          MOVL    #32/4,R4               ; SET ITERATION COUNT
            55    5E   D0        0740   698          MOVL    SP,R5                  ; SET POINTER INTO BUFFER
         FE AB    02   90        0743   699          MOVB    #2,CURTYPE-B(R11)      ; SET FOR LONGWORD FETCHES
                  6B   DD        0747   700          PUSHL   CURDOT-B(R11)          ; SAVE CURRENT LOCATION COUNTER
               FF38   30        0749   701 10$:      BSBW    FETCH                  ; FETCH LONGWORD
            85    04 AB   D0        074C   702          MOVL    QUAN-B(R11),(R5)+      ; STORE INTO INSTRUCTION BUFFER
               6B    04   C0        0750   703          ADDL    #4,CURDOT-B(R11)       ; SKIP TO NEXT LONGWORD
                  F3 54   F5        0753   704          SOBGTR  R4,10$                 ; FILL ENTIRE BUFFER
                  6B 8ED0   0756   705          POPL    CURDOT-B(R11)          ; RESTORE CURRENT LOCATION
                  55   DD        0759   706          PUSHL   R5                     ; ADDRESS OF DECODE OUTPUT BUFFER
                  32   DD        075B   707          PUSHL   #50                    ; LENGTH OF DECODE OUTPUT BUFFER
               9D'AF   9F        075D   708          PUSHAB  B^OUTPUT_ADDRESS       ; ADDRESS OF SYMBOLIZE ROUTINE
                  04 AE   3F        0760   709          PUSHAW  4(SP)                  ; ADDRESS OF WORD TO RECEIVE LENGTH
                  08 AE   7F        0763   710          PUSHAQ  8(SP)                  ; ADDRESS OF DECODE OUTPUT DESCRIPTOR
                  F8 AB   DF        0766   711          PUSHAL  INSBUF-B(R11)          ; ADDRESS OF INSTRUCTION STREAM POINTER
      00000000'GF   04   FB        0769   712          CALLS   #4,G^LIB$INS_DECODE    ; DECODE INSTRUCTION INTO BUFFER
               53   8E   7D        0770   713          MOVQ    (SP)+,R3               ; GET DESCRIPTOR OF STRING
                  1A 50   E9        0773   714          BLBC    R0,90$                 ; BRANCH IF ERROR DETECTED
                6443   94        0776   715          CLRB    (R4)[R3]               ; MAKE INTO ASCIZ STRING
                0169   30        0779   716          BSBW    OUTZSTRING             ; OUTPUT ASCIZ STRING
   F4 AB    F8 AB   5E   C3        077C   717          SUBL3   SP,INSBUF-B(R11),INSLEN-B(R11) ; SET LENGTH OF INSTRUCTION
      5E   00000052 8F   C0        0782   718 50$:      ADDL    #32+50,SP              ; DEALLOCATE STREAM/DECODE BUFFERS
            54    92 AF   9E        0789   719          MOVAB   SPACES,R4              ; SET ADDRESS OF SPACES
                0155   31        078D   720          BRW     OUTZSTRING             ; FOLLOW INSTRUCITON WITH SOME SPACE
                                0790   721 ;
                                0790   722 ; UNABLE TO DECODE INSTRUCTION (ACCVIO OR NEW INSTRUCTION).  OUTPUT LONGWORD
                                0790   723 ;
            53    F8 BB   D0        0790   724 90$:      MOVL    @INSBUF-B(R11),R3      ; GET FIRST LONGWORD OF STREAM
            F4 AB    01   D0        0794   725          MOVL    #1,INSLEN-B(R11)       ; SET INSTRUCTION LENGTH TO 1
                012D   30        0798   726          BSBW    OUTLONG                ; OUTPUT AS LONGWORD
                  E5   11        079B   727          BRB     50$
                                079D   728
                                079D   729 ;
                                079D   730 ; OUTPUT AN OPERAND WHICH IS A RELATIVE OR ABSOLUTE ADDRESS
                                079D   731 ;
                                079D   732
                                079D   733 OUTPUT_ADDRESS:
                      081C      079D   734          .WORD   ^M<R2,R3,R4,R11>
                                079F   735
            53    04 BC   D0        079F   736          MOVL    @4(AP),R3              ; GET VALUE (ARGUMENT BY REFERENCE)
            52    08 AC   D0        07A3   737          MOVL    8(AP),R2               ; GET ADDRESS OF DESCRIPTOR
              19 10 BC   E8        07A7   738          BLBS    @16(AP),5$             ; BRANCH IF ABSOLUTE ADDRESS
```

```
                        07AB   739          .IF     DF,SW_PROCESS              ; IF PROCESS VERSION,
              51   DC   07AB   740          MOVPSL  R1                         ; GET CURRENT PSL
51   51   02   18   EF   07AD   741          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R1,R1 ; ISOLATE CURRENT ACCESS MODE
  51   00E4 8F   A4   07B2   742          MULW    #CONTEXTS2,R1              ; COMPUTE OFFSET FROM KERNEL CONTEXT
5B   FBD0 CF41   9E   07B7   743          MOVAB   W^B[R1],R11                ; GET BASE ADDRESS OF CONTEXT AREA
                        07BD   744          .IFF                               ; FOR EXECUTIVE VERSION,
                        07BD   745          MOVAB   W^B,R11                    ; GET BASE ADDRESS OF CONTEXT AREA
                        07BD   746          .ENDC
        53   FB AB   C2   07BD   747          SUBL    INSBUF-B(R11),R3           ; GET OFFSET FROM INSTRUCTION
          53   6B   C0   07C1   748          ADDL    CURDOT-B(R11),R3           ; AND COMPUTE "REAL" ADDRESS
        04 AB   53   D0   07C4   749 5$:       MOVL    R3,QUAN-B(R11)             ; SET NEW "Q" SO THAT INDIRECTION
                        07C8   750                                             ; CAN BE USED TO SEE THE LAST OPERAND
                        07C8   751                                             ; OR BRANCHED-TO INSTRUCTION
           0C BC   B4   07C8   752          CLRW    @12(AP)                    ; ASSUME NOTHING PUT INTO BUFFER
        08   62   B1   07CB   753          CMPW    (R2),#8                    ; ENOUGH ROOM FOR 8 CHARACTERS?
             39   19   07CE   754          BLSS    20$                        ; BRANCH IF NOT ENOUGH
           04 A2   DD   07D0   755          PUSHL   4(R2)                      ; SAVE ADDRESS OF RESULT BUFFER
        51   53   D0   07D3   756          MOVL    R3,R1                      ; COPY EFFECTIVE ADDRESS
           0045   30   07D6   757          BSBW    RELOC                      ; SEE IF CLOSE TO RELOCATION REGISTER
             1F   19   07D9   758          BLSS    8$                         ; BRANCH IF NONE FOUND
        52   54   D0   07DB   759          MOVL    R4,R2                      ; SAVE OFFSET FROM X REGISTER
             54 8ED0   07DE   760          POPL    R4                         ; GET ADDRESS OF OUTPUT BUFFER
     84   5B 8F   90   07E1   761          MOVB    #^A'X',(R4)+               ; WRITE 'X'
             50   D4   07E5   762          CLRL    R0                         ; PRINT ONLY 1 DIGIT
             24   10   07E7   763          BSBB    100$                       ; WRITE REGISTER NUMBER
           84   2B   90   07E9   764          MOVB    #^A'+',(R4)+               ; WRITE '+'
           50   08   D0   07EC   765          MOVL    #8,R0                      ; PRINT 3 DIGITS
           53   52   D0   07EF   766          MOVL    R2,R3                      ; RETRIEVE OFFSET FROM X REGISTER
             19   10   07F2   767          BSBB    100$                       ; WRITE OFFSET IN HEX
        0C BC   06   B0   07F4   768          MOVW    #6,@12(AP)                 ; STORE LENGTH OF STRING
             0F   11   07F8   769          BRB     20$                        ; EXIT SUCCESSFULLY
          53   51   D0   07FA   770 8$:       MOVL    R1,R3                      ; GET EFFECTIVE ADDRESS
             54 8ED0   07FD   771          POPL    R4                         ; GET ADDRESS OF OUTPUT BUFFER
          50   1C   D0   0800   772          MOVL    #28,R0                     ; SET STARTING BIT FOR 1ST DIGIT
             08   10   0803   773          BSBB    100$                       ; OUTPUT HEX LONGWORD
        0C BC   08   B0   0805   774          MOVW    #8,@12(AP)                 ; RETURN LENGTH TO CALLER
          50   01   D0   0809   775 20$:       MOVL    #1,R0                      ; SUCCESS
                   04   080C   776          RET
                        080D   777
51   53   04   50   EF   080D   778 100$:     EXTZV   R0,#4,R3,R1                ; GET DIGIT
  84   FD02 CF41   90   0812   779          MOVB    PRIMARY[R1],(R4)+          ; MOVE DIGIT INTO BUFFER
          50   04   C2   0818   780          SUBL    #4,R0                      ; SKIP TO NEXT DIGIT
             F0   18   081B   781          BGEQ    100$                       ; LOOP UNTIL END OF LONGWORD
                   05   081D   782          RSB
```

```
                          081E   784          .SBTTL  DETERMINE CLOSEST RELOCATION REGISTER
                          081E   785 ;
                          081E   786 ; RELOC - GIVEN AN ADDRESS, RETURN CLOSEST RELOCATION REGISTER, IF ANY.
                          081E   787 ;
                          081E   788 ; INPUTS:
                          081E   789 ;
                          081E   790 ;       R1 = ADDRESS
                          081E   791 ;
                          081E   792 ; OUTPUTS:
                          081E   793 ;
                          081E   794 ;       X1 = EFFECTIVE ADDRESS
                          081E   795 ;       R3 = REGISTER #
                          081E   796 ;       R4 = OFFSET FROM X REGISTER
                          081E   797 ;       PSL CONDITION CODES SET ON R3
                          081E   798 ;       R2 DESTROYED.
                          081E   799 ;
          53    D4      081E   800 RELOC:  CLRL    R3                      ; START WITH X0
       52   01    CE    0820   801         MNEGL   #1,R2                    ; X REGISTER CLOSEST TO ADDRESS
       54   52    3C    0823   802         MOVZWL  R2,R4                   ; CLOSEST SO FAR IS FFFF
 50  FBE2 CF43   D0    0826   803 10$:     MOVL    XREGV[R3],R0            ; GET X REGISTER
          18    13    082C   804         BEQL    15$                     ; BRANCH IF NOT VALID
    50   51    50    C3    082E   805         SUBL3   R0,R1,R0                ; GET OFFSET FROM X#
00000800 8F    50    D1    0832   806         CMPL    R0,#^X800               ; WITHIN REASONABLE RANGE?
          08    1E    0839   807         BGEQU   15$                     ; BRANCH IF OK
       54   50    D1    083B   808         CMPL    R0,R4                   ; CLOSER THAN CLOSEST SO FAR?
          06    1A    083E   809         BGTRU   15$                     ; BRANCH IF NOT
       52   53    D0    0840   810         MOVL    R3,R2                   ; SAVE X# CLOSEST TO ADDRESS
       54   50    D0    0843   811         MOVL    R0,R4                   ; AND SET NEW CLOSEST OFFSET
    DC 53   08    F2    0846   812 15$:     AOBLSS  #8,R3,10$               ; LOOP UNTIL LAST REGISTER TESTED
       53   52    D0    084A   813         MOVL    R2,R3                   ; RETURN X# CLOSEST TO ADDRESS
                05    084D   814         RSB
```

```
                     084E    816          .SBTTL  OUTPUTA - OUTPUT ADDRESS
                     084E    817 ;
                     084E    818 ;       OUTPUT ADDRESS
                     084E    819 ;
                     084E    820 OUTPUTA:                                    ; OUTPUT ADDRESS
        0140    30   084E    821          BSBW    CRLF                       ; OUTPUT CR/LF
    >1    6B    D0   0851    822          MOVL    CURDOT-B(R11),R1           ; GET ADDRESS
          C8    10   0854    823          BSBB    RELOC                      ; SEE IF CLOSE TO RELOCATION REGISTER
          1B    19   0856    824          BLSS    2$                         ; BRANCH IF NOT
          54    DD   0858    825          PUSHL   R4                         ; SAVE OFFSET FROM RELOCATION REGISTER
    50   58 8F   9A   085A    826          MOVZBL  #^A'X',R0                  ; OUTPUT AN 'X'
         00E1    30   085E    827          BSBW    OUTCHAR
          52    D4   0861    828          CLRL    R2                         ; PRINT ONLY 1 HEX DIGIT
          66    10   0863    829          BSBB    OUTCOM                     ; OUTPUT HEX VALUE IN R3
    50    2B    9A   0865    830          MOVZBL  #^A'+',R0                  ; OUTPUT AN '+'
         00D7    30   0868    831          BSBW    OUTCHAR
    52    08    D0   0868    832          MOVL    #8,R2                      ; PRINT 3 DIGITS
          53 8ED0   086E    833          POPL    R3                         ; GET OFFSET FROM RELCATION REGISTER
          49    11   0871    834          BRB     10$                        ; OUTPUT OFFSET AND SLASH
    53   18 AB    9E   0873    835 2$:      MOVAB   SAVREG-B(R11),R3           ; BASE OF REGISTER AREA
                     0877    836          .IF     DF,SW_PROCESS              ; ONLY FOR PROCESS VERSION
         F0 AB    D5   0877    837          TSTL    PID-B(R11)                 ; CHECK FOR OTHER PROCESS ADDRESS
          19    12   087A    838          BNEQ    3$                         ; BR IF YES
                     087C    839          .ENDC
53   6B   53    C3   087C    840          SUBL3   R3,CURDOT-B(R11),R3        ; COMPUTE OFFSET INTO REGISTER AREA
          26    19   0880    841          BLSS    5$                         ; NOT GENERAL REGISTER
    53    04    C6   0882    842          DIVL    #4,R3                      ; SCALE TO LONGWORD NUMBER
    0F    53    D1   0885    843          CMPL    R3,#15                     ; CHECK FOR MAX REG NUMBER
          1E    14   0888    844          BGTR    5$                         ; GTR, NOT A REGISTER
    50   52 8F   9A   088A    845          MOVZBL  #^A'R',R0                  ; OUTPUT PREFIX
         00B1    30   088E    846          BSBW    OUTCHAR                    ; OF 'R'
          52    D4   0891    847          CLRL    R2                         ; AND SET FOR ONE DIGIT OF OUTPUT
          27    11   0893    848          BRB     10$
                     0895    849          .IF     DF,SW_PROCESS              ; FOR PROCESS VERSION ONLY
          6A    D5   0895    850 3$:      TSTL    (R10)                      ; CHECK FOR PROCESSOR REGISTER
          0F    19   0897    851          BLSS    5$                         ; BR IF YES
    52    1C    D0   0899    852          MOVL    #28,R2                     ; SET FOR LONGWORD OUTPUT
53   F0 AB    D0   089C    853          MOVL    PID-B(R11),R3              ; GET PID OF TARGET
          29    10   08A0    854          BSBB    OUTCOM                     ; OUTPUT PID AS LONGWORD
    50    3A    9A   08A2    855          MOVZBL  #^A':',R0                  ; SEPARATE WITH ':'
         009A    30   08A5    856          BSBW    OUTCHAR                    ; OUTPUT COLON
                     08A8    857          .ENDC
    53    6B    D0   08A8    858 5$:      MOVL    CURDOT-B(R11),R3           ; GET ADDRESS
    52    1C    D0   08AB    859          MOVL    #28,R2                     ; ASSUME LONGWORD OUTPUT
          6A    D5   08AE    860          TSTL    (R10)                      ; CHECK FOR PROCESSOR REGISTER
          0A    18   08B0    861          BGEQ    10$                        ; NO, JUST A LONGWORD
50   50 8F   9A   08B2    862          MOVZBL  #^A'P',R0                  ; PRECEDE WITH A 'P'
         0089    30   08B6    863          BSBW    OUTCHAR                    ; OUTPUT P
    52    04    D0   08B9    864          MOVL    #4,R2                      ; SET FIELD TO 2 DIGITS
          0D    10   08BC    865 10$:     BSBB    OUTCOM                     ; COMMON OUTPUT
    50    2F    9A   08BE    866          MOVZBL  #SLSH,R0                   ; OUTPUT SLASH
         007E    31   08C1    867          BRW     OUTCHAR                    ; RETURN THROUGH OUTCHAR
                     08C4    868 OUTDIGIT:                                   ; OUTPUT ONE DIGIT
          52    D4   08C4    869          CLRL    R2                         ; ZAP DIGIT SELECTOR
          03    11   08C6    870          BRB     OUTCOM                     ; AND MERGE WITH COMMON
                     08C8    871
                     08C8    872 OUTLONG:                                    ; OUTPUT LONGWORD
```

```
                52   1C   D0  08C8   873                MOVL    #28,R2                  ; SET DIGIT SELECTOR
                                08CB   874 OUTCOM:                                     ; FORMAT IT
          54    08   AB   9E  08CB   875                MOVAB   OUTBUF-B(R11),R4        ; GET ADDRESS OF OUTPUT BUFFER
51   53    04        52   EF  08CF   876 10$:           EXTZV   R2,#4,R3,R1             ; GET DIGIT
     84  FC40  CF41        90  08D4   877                MOVB    PRIMARY[R1],(R4)+       ; BUFFER IT
                52   04   C2  08DA   878                SUBL    #4,R2                   ; NEXT DIGIT
                     F0   18  08DD   879                BGEQ    10$                     ; DO ALL REQUESTED
                     64   94  08DF   880                CLRB    (R4)                    ; MARK END OF BUFFER
          54    08   AB   9E  08E1   881 OUTZBUF:       MOVAB   OUTBUF-B(R11),R4        ; GET START OF BUFFER
                                08E5   882
                                08E5   883 OUTZSTRING:                                 ; OUTPUT ASCIZ STRING
                                08E5   884                .IF     NDF,SW_PROCESS
                                08E5   885                MOVZBL  (R4)+,R0                ; GET A CHAR
                                08E5   886                BEQL    10$                     ; BR IF DONE
                                08E5   887                BSBB    OUTCHAR                 ; OUTPUT CHAR
                                08E5   888                BRB     OUTZSTRING              ; CONTINUE
                                08E5   889 10$:           RSB                             ; RETURN IF DONE
                                08E5   890                .IFF
                     55   DD  08E5   891                PUSHL   R5                      ; Save a register.
64   0100  8F        00   3A  08E7   892                LOCC    #0, #256, (R4)          ; Locate the terminating zero.
     55    51        54   C3  08ED   893                SUBL3   R4, R1, R5              ; Compute the number of bytes to write.
                     42   13  08F1   894                BEQL    90$                     ; Branch if zero bytes to write.
                                08F3   895 50$:           $QIOW_S EFN=#30,-               ; Write whole buffer.
                                08F3   896                        CHAN=TTCHAN,-
                                08F3   897                        FUNC=#IO$_WRITEVBLK,-
                                08F3   898                        P1=(R4),-
                                08F3   899                        P2=R5
0124  8F      50   B1  0910   900                CMPW    R0,#SS$_INSFMEM         ; If any resource error occurs,
               13   13  0915   901                BEQL    60$                     ; wait for an I/O completion
        1C     50   B1  0917   902                CMPW    R0,#SS$_EXQUOTA         ; and try again.
               0E   13  091A   903                BEQL    60$
2A00  8F      50   B1  091C   904                CMPW    R0,#SS$_EXQUOTASTRT
               12   1F  0921   905                BLSSU   90$
2AFF  8F      50   B1  0923   906                CMPW    R0,#SS$_EXQUOTAEND
               08   1A  0928   907                BGTRU   90$
                         092A   908 60$:           $WAITFR_S  EFN=#30
                     BE   11  0933   909                BRB     50$
                55   BED0  0935   910 90$:           POPL    R5                      ; Restore saved register.
                          05  0938   911                RSB
                              0939   912                .ENDC
                              0939   913
                              0939   914
                              0939   915 OUTBSLSH:                                  ; OUTPUT BACK SLASH
          50    5C  8F   9A  0939   916                MOVZBL  #BSLSH,R0               ; SET CHARACTER CODE
                03   11  093D   917                BRB     OUTCHAR                 ; AND OUTPUT IT
     50         58   9A  093F   918 OUTR8:   MOVZBL  R8,R0                          ; GET CHAR TO OUTPUT
                              0942   919 OUTCHAR:                                   ; OUTPUT CHAR IN R0
                              0942   920                .IF     NDF,SW_PROCESS
                              0942   921                TSTL    AP                      ; CHECK FOR CONSOLE
                              0942   922                BNEQ    10$                     ; NO, USE DEVICE DIRECTLY
                              0942   923                JMP     G^CON$PUTCHAR           ; OUTPUT TO THE CONSOLE TERMINAL
                              0942   924
                              0942   925 10$:           MOVW    OUTCR(AP),R1            ; GET STATUS
                              0942   926                BBC     #7,R1,10$               ; WAIT FOR READY
                              0942   927                MOVB    R0,OUTB(AP)             ; OUTPUT CHAR
                              0942   928                .IFF                            ; FALSE FOR PROCESS VERSION
                     50   DD  0942   929                PUSHL   R0                      ; BUFFER CHARACTER ON STACK
```

```
                50    5E    D0  0944    930 50$:    MOVL    SP,R0                   ; SAVE POINTER TO IT
                                  0947    931         $QIOW_S EFN=#30,-               ;
                                  0947    932                 CHAN=TTCHAN,-           ;
                                  0947    933                 FUNC=#IO$_WRITEVBLK,-   ;
                                  0947    934                 P1=(R0),-               ; BUFFER ADDRESS
                                  0947    935                 P2=#1                   ; ONE CHARACTER
          0124  8F    50    B1  0964    936         CMPW    R0,#SS$_INSFMEM         ; If any resource error occurs,
                      13    13  0969    937         BEQL    60$                     ; wait for an I/O completion
                1C    50    B1  096B    938         CMPW    R0,#SS$_EXQUOTA         ; and try again.
                      0E    13  096E    939         BEQL    60$
          2A00  8F    50    B1  0970    940         CMPW    R0,#SS$_EXQUOTASTRT
                      12    1F  0975    941         BLSSU   90$
          2AFF  8F    50    B1  0977    942         CMPW    R0,#SS$_EXQUOTAEND
                      0B    1A  097C    943         BGTRU   90$
                                  097E    944 60$:    $WAITFR_S   EFN=#30
                      BB    11  0987    945         BRB     50$
                      01    BA  0989    946 90$:    POPR    #^M<R0>                 ; RESTORE CHARACTER
                                  098B    947         .ENDC                           ;
                      05        098B    948         RSB                             ; AND RETURN
                                  098C    949 OUTSPACE:                               ;
                50    20    9A  098C    950         MOVZBL  #32,R0                  ; SET CODE FOR SPACE
                      B1    11  098F    951         BRB     OUTCHAR                 ; AND SEND IT
                50    0D    9A  0991    952 CRLF:   MOVZBL  #CR,R0                  ; RETURN
                      AC    10  0994    953         BSBB    OUTCHAR                 ; SEND IT
                50    0A    9A  0996    954         MOVZBL  #LF,R0                  ; LINE FEED
                      A7    11  0999    955         BRB     OUTCHAR                 ; SEND IT
                                  099B    956
                                  099B    957
```

```
                     099B   959          .SBTTL  GETCHAR - GET INPUT CHARACTER ROUTINE
                     099B   960
                     099B   961 ;
                     099B   962 ;         GETCHAR - GET INPUT CHARACTER
                     099B   963 ;
                     099B   964 ; OUTPUT:
                     099B   965 ;         R8 - INPUT CHARACTER
                     099B   966 ;         R9 - BUFFER POINTER UPDATED (BUFFER IN ASCIZ FORMAT)
                     099B   967 ;
                     099B   968
                     099B   969 GETCHAR:                                             ;
          58   89   9A  099B   970          MOVZBL  (R9)+,R8                        ; GET NEXT CHARACTER
               01   13  099E   971          BEQL    10$                             ; READ IF NONE AVAIL
                    05  09A0   972          RSB
        59   84 AB   9E  09A1   973 10$:     MOVAB   INBUF-B(R11),R9                 ; SET ADDRESS OF INPUT BUFFER
                     09A5   974          .IF     NDF,SW_PROCESS                      ;
                     09A5   975 20$:     TSTL    AP                              ; CHECK FOR CONSOLE
                     09A5   976          BNEQ    30$                             ; YES
                     09A5   977          JSB     G^CON$GETCHAR                   ; GET A CHARACTER FROM THE CONSOLE TERMINAL
                     09A5   978          MOVB    R0,R8                           ;
                     09A5   979          BRB     60$                             ; CONTINUE IN COMMON
                     09A5   980 30$:     MOVW    RDCR(AP),R0                     ; GET STATUS
                     09A5   981 40$:     BBC     #7,R0,30$                       ; WAIT FOR READY
                     09A5   982          MOVB    RDBUF(AP),R8                    ; GET CHARACTER
                     09A5   983          BRB     60$                             ; MERGE WITH COMMON
                     09A5   984          .IFF                                    ; FALSE IF PROCESS VERSION
        50   FABA CF   DE  09A5   985 15$:     MOVAL   TTITMLST,R0                     ; get the relocateable address
                     09AA   986          $QIOW_S EFN=#31,-                       ;
                     09AA   987                  CHAN=TTCHAN,-                   ; INPUT DEVICE CHANNEL
                     09AA   988                  IOSB=TTIOSB,-                   ; IO STATUS BLOCK
                     09AA   989                  FUNC=#<IO$_READVBLK!IO$M_EXTEND>,-      ;
                     09AA   990                  P1=(R9),-                       ; BUFFER ADDRESS
                     09AA   991                  P2=#80,-                        ; READ SIZE
                     09AA   992                  P5=R0,-
                     09AA   993                  P6=#TTITMLSTLEN
  0124 8F    50   B1  09D1   994          CMPW    R0,#SS$_INSFMEM                 ; If any resource error occurs,
         13   13  09D6   995          BEQL    760$                            ; wait for an I/O completion
       1C    50   B1  09D8   996          CMPW    R0,#SS$_EXQUOTA                 ; and try again.
         0E   13  09DB   997          BEQL    760$
  2A00 8F    50   B1  09DD   998          CMPW    R0,#SS$_EXQUOTASTRT
         12   1F  09E2   999          BLSSU   790$
  2AFF 8F    50   B1  09E4  1000          CMPW    R0,#SS$_EXQUOTAEND
         0B   1A  09E9  1001          BGTRU   790$
                     09EB  1002 760$:    $WAITFR_S   EFN=#31
              AF   11  09F4  1003          BRB     15$
                     09F6  1004 790$:
    50   FA55 CF   3C  09F6  1005          MOVZWL  TTIOSB+2,R0                     ; GET SIZE READ
  8049   FA52 CF   90  09FB  1006          MOVB    TTIOSB+4,(R0)+[R9]              ; BUFFER TERMINATOR
          6940   94  0A01  1007          CLRB    (R9)[R0]                        ; MARK END OF BUFFER
         52   59   D0  0A04  1008          MOVL    R9,R2                           ; POINT TO START OF STRING
         58   82   9A  0A07  1009 20$:     MOVZBL  (R2)+,R8                        ; GET A CHARACTER
              99   13  0A0A  1010          BEQL    15$                             ; EMPTY, READ SOME MORE
                     0A0C  1011          .ENDC                                   ;
     58   80 8F   8A  0A0C  1012 60$:     BICB    #^X80,R8                        ; STRIP PARITY
     7F 8F    58   91  0A10  1013          CMPB    R8,#RUBOUT                      ; CHECK FOR RUBOUT
              15   12  0A14  1014          BNEQ    90$                             ; NO
     03 6A    06   E2  0A16  1015          BBSS    #V_RUB,(R10),70$                ; SET START OF RUBOUT SEQUENCE
```

```
                FF1C   30   0A1A  1016          BSBW    OUTBSLSH                ; OUTPUT BACK SLASH
             58   79   9A   0A1D  1017 70$:     MOVZBL  -(R9),R8                ; GET RUBBED OUT CHAR
                  04   12   0A20  1018          BNEQ    80$                     ; SKIP INC
                  59   D6   0A22  1019          INCL    R9                      ; POINT AT START OF BUFFER
                  E1   11   0A24  1020          BRB     20$                     ; AND GET ANOTHER
                FF16   30   0A26  1021 80$:     BSBW    OUTRB                   ; OUTPUT RUBBED OUT CHAR
                  DC   11   0A29  1022          BRB     20$                     ; AND GET ANOTHER
          03 6A    06   E5   0A2B  1023 90$:     BBCC    #V_RUB,(R10),100$       ; TERMINATE RUBOUT SEQUENCE
                FF07   30   0A2F  1024          BSBW    OUTBSLSH                ; OUTPUT BACK SLASH
          03 58    06   E1   0A32  1025 100$:    BBC     #6,R8,110$              ; BR IF NOT ALPHA
             58   20   8A   0A36  1026          BICB    #32,R8                  ; SET TO UPPER CASE
                           0A39  1027 110$:
                           0A39  1028          .IF     NDF,SW_PROCESS          ;
                           0A39  1029          BSBW    OUTR8                   ; ECHO CHARACTER
                           0A39  1030          .ENDC                           ;
          89   58   90   0A39  1031          MOVB    R8,(R9)+                ; BUFFER NEW CHAR
FAF0 CF   0A   58   3A   0A3C  1032          LOCC    R8,#NTERM,TERM          ; CHECK FOR TERMINATOR
                  C3   13   0A42  1033          BEQL    20$                     ; NOT A TERMINATOR
             58   0D   91   0A44  1034          CMPB    #CR,R8                  ; IS CHAR = RETURN
                  03   12   0A47  1035          BNEQ    120$                    ; NO,
                FF45   30   0A49  1036          BSBW    CRLF                    ; YES, SEND CR/LF
                  69   94   0A4C  1037 120$:     CLRB    (R9)                    ; MARK END OF BUFFER
          59   84 AB   9E   0A4E  1038          MOVAB   INBUF-B(R11),R9         ; RESTORE BUFFER BASE
                FF46   31   0A52  1039          BRW     GETCHAR                 ; AND TRY AGAIN
```

```
                        0A55  1041          .SBTTL  PLUS/MINUS OPERATORS
                        0A55  1042 ;
                        0A55  1043 ;       PLUS/MINUS OPERATORS
                        0A55  1044 ;
                        0A55  1045 BLANK:                                       ; SAME AS PLUS
                        0A55  1046 OPERATOR:                                    ;
              FB83   30 0A55  1047          BSBW    ENDEXPR                   ; END EXPR
FF AB   50     12   83 0A58  1048          SUBB3   #OPERBAS,R0,OPER-B(R11)   ; SET OPERATOR
                     05 0A5D  1049          RSB                               ; RETURN
                        0A5E  1050 ;
                        0A5E  1051 ;       MONADIC MINUS - NEGATE
                        0A5E  1052 ;
                  56    D5 0A5E  1053 NEGATE: TSTL    R6                        ; TEST ACCUMULATOR
                  03    13 0A60  1054          BEQL    5$                        ; EMPTY
                FB76    30 0A62  1055          BSBW    ENDEXPR                   ; OTHERWISE PERFORM OPERATION
           6A   80  8F    8C 0A65  1056 5$:     XORB    #<1@V_NEGATE>,(R10)       ; TOGGLE NEGATE FLAG
                     05 0A69  1057 10$:    RSB                               ; AND RETURN
                        0A6A  1058
                        0A6A  1059
```

```
                                0A6A  1061          .SBTTL  TAB - INDIRECT DISPLAY
                                0A6A  1062 ;
                                0A6A  1063 ;       TAB
                                0A6A  1064 ;
        6B   04 AB      D0      0A6A  1065 TAB:    MOVL    QUAN-B(R11),CURDOT-B(R11)       ; GO INDIRECT
50  6A   01    0F      EF      0A6E  1066         EXTZV   #V_PRMODE,#1,(R10),R0    ; GET PROCESSOR REGISTER MODE
6A  01   1F    50      F0      0A73  1067         INSV    R0,#V_PREG,#1,(R10)     ; AND COPY TO SEMI-PERMANENT FLAG
               13      11      0A78  1068         BRB     LOCP                    ; AND DISPLAY IT
                              0A7A  1069
                              0A7A  1070 ;
                              0A7A  1071 ;       ESCAPE - DISPLAY PREVIOUS LOCATION
                              0A7A  1072 ;
                              0A7A  1073
                              0A7A  1074 ESCAP:                                  ;
     0F 6A     0D      E0      0A7A  1075         BBS     #V_INSTR,(R10),LOCP     ; BRANCH IF INSTRUCTION MODE
        51     01      D0      0A7E  1076         MOVL    #1,R1                   ; ASSUME UNIT INCREMENT
               6A      D5      0A81  1077         TSTL    (R10)                   ; CHECK FOR PROCESSOR REGISTER
               05      19      0A83  1078         BLSS    10$                     ; YES, USE UNIT INCREMENT
51  51     FE AB      9C      0A85  1079         ROTL    CURTYPE-B(R11),R1,R1    ; FORM INCREMENT
        6B     51      C2      0A8A  1080 10$:    SUBL    R1,CURDOT-B(R11)        ; AND SUBTRACT FROM DOT
             FC5A      31      0A8D  1081 LOCP:   BRW     LOCPROMPT               ; PROMPT WITH CONTENT
```

```
                        0A90  1083          .SBTTL  DISPLAY INSTRUCTION RANGE
                        0A90  1084 ;
                        0A90  1085 ;       'I' - DISPLAY RANGE OF INSTRUCTIONS
                        0A90  1086 ;
      FBD6   30         0A90  1087 INSTR:  BSBW    ENDFIELD                    ; TERMINATE FIELD
   6A   02   8A         0A93  1088         BICB    #1@V_ASCII,(R10)            ; CLEAR CHARACTER DISPLAY MODE
06 6A   08   E0         0A96  1089         BBS     #V_FT,(R10),5$              ; ADDRESS SPECIFIED?
6B   04 AB   D0         0A9A  1090         MOVL    QUAN-B(R11),CURDOT-B(R11) ; EXAMINE AT Q IF UNSPECIFIED
        04   11         0A9E  1091         BRB     10$
6B   D8 AB   D0         0AA0  1092 5$:     MOVL    F1-B(R11),CURDOT-B(R11) ; IF ADDRESS SPECIFIED, SET NEW DOT
6A 2000 8F   A8         0AA4  1093 10$:    BISW    #1@V_INSTR,(R10)            ; SET INSTRUCTION DISPLAY MODE
      FC76   30         0AA9  1094         BSBW    OUTINS                      ; DISPLAY INSTRUCTION
0B 6A   09   E1         0AAC  1095         BBC     #V_F2,(R10),30$             ; IF NO RANGE SPECIFIED, EXIT
6B   DC AB   D1         0AB0  1096 20$:    CMPL    F2-B(R11),CURDOT-B(R11) ; END OF RANGE?
        05   15         0AB4  1097         BLEQ    30$                         ; BRANCH IF DONE
      FC2F   30         0AB6  1098         BSBW    NEXTLOC                     ; OUTPUT NEXT INSTRUCTION
        F5   11         0AB9  1099         BRB     20$                         ; LOOP UNTIL DONE
        0F   11         0ABB  1100 30$:    BRB     RESET                       ; RESET SCANNER
```

```
                        0ABD  1102          .SBTTL  EQUALS - DISPLAY VALUE
                        0ABD  1103 ;
                        0ABD  1104 ;       EQUALS - VALUE DISPLAY
                        0ABD  1105 ;
                        0ABD  1106 EQUAL5:                                  ;
                        0ABD  1107          .ENABL  LSB                     ;
                FBA9    30    0ABD  1108          BSBW    ENDFIELD          ; TERMINATE FIELD
             05 6A  08  E1    0AC0  1109          BBC     #V_F1,(R10),10$   ; IGNORE IF FIELD BLANK
          04 AB  DB AB  D0    0AC4  1110 EQL1:    MOVL    F1=B(R11),QUAN-B(R11) ; SET QUANTITY
                FC2A    30    0AC9  1111 10$:     BSBW    OUTPUT            ; OUTPUT IT
                        0ACC  1112 ;        BRB     RESET             ; RESET SCANNER
                        0ACC  1113          .DSABL  LSB                     ;
                        0ACC  1114
                        0ACC  1115 ;
                        0ACC  1116 ;       RESET
                        0ACC  1117 ;
                        0ACC  1118
    6A  00FFDF80  8F    CA    0ACC  1119 RESET:   BICL    #^XOFFDF80,(R10)  ; CLEAR FIELD AND NEGATE FLAGS
                FC  AB  94    0AD3  1120          CLRB    FCTR-B(R11)       ; CLEAR FIELD COUNTER
                    56  7C    0AD6  1121          CLRQ    R6                ; RESET ACCUMULATORS
                        05    0ADB  1122          RSB                       ; RETURN
```

```
                        0AD9  1124          .SBTTL  SEMI - SECONDARY COMMAND SET
                        0AD9  1125 ;
                        0AD9  1126 ;       SEMI
                        0AD9  1127 ;
                        0AD9  1128
                        0AD9  1129 SECOND:                                      ;
                     58 0AD9  1130          .ASCII  /X/                         ; X REGISTER SET/DISPLAY
                     50 0ADA  1131          .ASCII  /P/                         ; P - PROCEED
                     4D 0ADB  1132          .ASCII  /M/                         ; M - SET MODIFY FLAG
                     49 0ADC  1133          .ASCII  /I/                         ; I - PROGRAM COUNTER
                     47 0ADD  1134          .ASCII  /G/                         ; G - GO, START
                     45 0ADE  1135          .ASCII  /E/                         ; E - EXECUTE STRING
                     42 0ADF  1136          .ASCII  /B/                         ; B - SET/CLR BREAKPOINT
              00000007  0AE0  1137 NSEC=.-SECOND                                ; NUMBER OF SECONDARY COMMANDS
                        0AE0  1138
                        0AE0  1139 SEMI:                                        ;
           6A   01   8A 0AE0  1140          BICB    #<1@V_OPEN>,(R10)           ; CLEAR OPEN FLAG
              FB83   30 0AE3  1141          BSBW    ENDFIELD                    ; TERMINATE FIELD
              FEB2   30 0AE6  1142          BSBW    GETCHAR                     ; GET SECONDARY COMMAND CHAR
        EB AF 07   58   3A 0AE9  1143          LOCC    R8,#NSEC,SECOND             ; LOCATE SECONDARY COMMAND
                        0AEE  1144 10$:       CASE    R0,LIMIT=#1,<-              ; SWITCH ON TYPE
                        0AEE  1145                    BRKPOINT,-                  ; SET BREAKPOINT
                        0AEE  1146                    EXECUTE,-                   ; EXECUTE STRING
                        0AEE  1147                    GO,-                        ; SEMI-G, GO
                        0AEE  1148                    PROGCTR,-                   ; SEMI-I, INSTRUCTION CONTER
                        0AEE  1149                    MFYFLGS,-                   ; SEMI-M, MODIFY FLAG
                        0AEE  1150                    PROCED,-                    ; SEMI-P, PROCEED
                        0AEE  1151                    XSET,-                      ; SET XREGISTER
                        0AEE  1152                    >                           ;
              FA53   31 0B00  1153 ERR2:    BRW     ERROR                       ; ERROR
```

```
                                0803  1155          .SBTTL  LEFT BRACKET - MODE SELECTION
                                0803  1156 ;
                                0803  1157 ;
                                0803  1158 ;       LEFT BRACKET
                                0803  1159 ;
                                0803  1160 MODES:                                ; MODE CHARACTER LIST
                             49 0803  1161          .ASCII  /I/                 ; INSTRUCTION MODE, VARIABLE LENGTH
                             43 0804  1162          .ASCII  /C/                 ; CHARACTER, CURRENT LENGTH
                             4C 0805  1163          .ASCII  /L/                 ; LONG, HEX
                             57 0806  1164          .ASCII  /W/                 ; WORD, HEX
                             42 0807  1165          .ASCII  /B/                 ; BYTE, HEX
                     00000005 0808  1166 NMODES=.-MODES                          ; NUMBER OF MODE CHARACTERS
                                0808  1167
                                0808  1168
                                0808  1169 LBRACKET:                             ; MODE SELECTION
                         FE90   30 0808  1170          BSBW    GETCHAR             ; GET MODE CHAR
F3 AF    05      58         3A 0808  1171          LOCC    R8,#NMODES,MODES    ; CONVERT TO INDEX
                EE              0810  1172          BEQL    ERR2                ; NOT FOUND, ERROR
         04     50         D1 0812  1173          CMPL    R0,#4               ; CHECK FOR 'C'
                10              0815  1174          BEQL    10$                 ; BRANCH IF 'C'
                17              0817  1175          BGTR    20$                 ; BRANCH IF 'I'
FE AB    50     01         83 0819  1176          SUBB3   #1,R0,CURTYPE-B(R11)    ; SET MODE
   6A  2000 8F    AA 081E  1177          BICW    #1@V_INSTR,(R10)            ; CLEAR INSTRUCTION DISPLAY MODE
         6A     02      8A 0823  1178 5$:       BICB    #<1@V_ASCII>,(R10)          ; CLEAR CHARACTER DISPLAY MODE
                05      0826  1179          RSB                                 ; RETURN
         6A     02      88 0827  1180 10$:      BISB    #<1@V_ASCII>,(R10)          ; SET CHARACTER DISPLAY MODE
   6A  2000 8F    AA 082A  1181          BICW    #1@V_INSTR,(R10)            ; CLEAR CHARACTER DISPLAY MODE
                05      082F  1182          RSB
   6A  2000 8F    A8 0830  1183 20$:      BISW    #1@V_INSTR,(R10)            ; SET INSTRUCTION DISPLAY MODE
                EC   11 0835  1184          BRB     5$
```

```
                        0837  1186          .SBTTL  SINGLE STEP
                        0837  1187 ;
                        0837  1188 ;       STEP
                        0837  1189 ;
      6A  02    03  01  F0   0837  1190 STEP:   INSV    #1,#V_TBIT,#2,(R10)     ; CLR V_ATBRK, SET V_TBIT
       6A   80008000 8F  CA   083C  1191         BICL    #<<1@V_PRMODE>!<1@V_PREG>>,(R10)        ; CLEAR PROCESSOR REGISTER M
                        04   0843  1192         RET                             ; AND RETURN
```

```
                        0844  1194          .SBTTL  STEPOVER - STEP OVER ROUTINE CALL
                        0844  1195 ;
                        0844  1196 ;       STEPOVER
                        0844  1197 ;
                        0844  1198 STEPOVER:
      50    54 BB   9A  0844  1199          MOVZBL  @SAVPC-B(R11),R0        ; GET NEXT INSTRUCTION TO EXECUTE
6A  80008000 8F   CA  0848  1200          BICL    #<<1@V_PRMODE>!<1@V_PREG>>,(R10)        ; CLEAR PROCESSOR REGISTER M
   51   F9C1 CF   9E  084F  1201          MOVAB   OVEROPCODES,R1          ; ADDRESS OF LIST OF OPCODES
        52    05   9A  0854  1202          MOVZBL  #OVEROPCLEN,R2          ; SIZE OF TABLE
        81    50   91  0857  1203 10$:     CMPB    R0,(R1)+                ; MATCH?
              05   13  085A  1204          BEQL    20$                     ; BRANCH IF FOUND
           F8 52   F5  085C  1205          SOBGTR  R2,10$                  ; LOOP UNTIL FOUND
              D6   11  085F  1206          BRB     STEP                    ; IF NOT A ROUTINE CALL, NORMAL STEP
52  00000000'GF   9E  0861  1207 20$:     MOVAB   G^LIB$INS_DECODE,R2     ; GET ADDRESS OF FOLLOWING INSTRUCTION
              28   13  0868  1208          BEQL    30$                     ; IF NOT AVAILABLE, ERROR
           54 AB   DD  086A  1209          PUSHL   SAVPC-B(R11)            ; COPY ADDRESS OF INSTRUCTION STREAM
              00   DD  086D  1210          PUSHL   #0                      ; PUSH NULL DESCRIPTOR
              5E   DD  086F  1211          PUSHL   SP                      ; ADDRESS OF OUTPUT DESCRIPTOR
           08 AE   DF  0871  1212          PUSHAL  8(SP)                   ; ACCESS INSTRUCTION STREAM DIRECTLY
        62    02   FB  0874  1213          CALLS   #2,(R2)                 ; FIND ADDRESS OF FOLLOWING INSTRUCTION
           15 50   E9  0877  1214          BLBC    R0,25$                  ; IF NOT INTERPRETABLE, ERROR
                       087A  1215          .IF     DF,SW_PROCESS           ;
        54    04 AE   D0  087A  1216          MOVL    4(SP),R4                ; GET ADDRESS OF NEXT INSTRUCTION
        55    54   D0  087E  1217          MOVL    R4,R5                   ; MAKE END=START
           083F   30  0881  1218          BSBW    SETWRT                  ; MAKE INSTRUCTION WRITABLE
                       0884  1219          .ENDC
              03   BA  0884  1220          POPR    #^M<R0,R1>              ; GET UPDATED STREAM POINTER
        61    61   90  0886  1221          MOVB    (R1),(R1)               ; ERROR IF UNABLE TO WRITE BREAKPOINT
   F832 CF    51   D0  0889  1222          MOVL    R1,OVRADR               ; SET TEMPORARY BREAKPOINT
              04   0B8E  1223          RET                             ; START EXECUTION
                   0B8F  1224
   5E    08   C0  0B8F  1225 25$:     ADDL    #8,SP                   ; CLEAN STACK
      F9C1   31  0B92  1226 30$:     BRW     ERROR                   ; REPORT ERROR - UNABLE TO STEP OVER
```

```
                                0B95  1228          .SBTTL  BRKPOINT - SET/CLEAR BREAKPOINTS
                                0B95  1229 ;
                                0B95  1230 ;       BRKPOINT
                                0B95  1231 ;
                                0B95  1232 BRKPOINT:
          6C 6A    08   E1     0B95  1233          BBC     #V_F1,(R10),$HOBRK      ;DISPLAY BREAKPOINTS
          13 6A    09   E0     0B99  1234          BBS     #V_F2,(R10),20$         ;YES, IT WAS SPECIFIED
             52    01   D0     0B9D  1235          MOVL    #1,R2                   ;INIT INDEX
           F7F7 CF42   D5     0BA0  1236 10$:      TSTL    BRKADR[R2]              ;FIND FREE SLOT
                 14    13     0BA5  1237          BEQL    30$                     ;YES, GOT ONE
      FFF3 52    01   08   F1     0BA7  1238          ACBL    #NBRK,#1,R2,10$         ;CHECK THEM ALL
              F9A6    31     0BAD  1239 15$:      BRW     ERROR                   ;ERROR
          52    DC AB   D0     0BB0  1240 20$:      MOVL    F2-B(R11),R2            ;GET BRKPOINT NUMBER
                 EA    13     0BB4  1241          BEQL    10$                     ;NULL FIELD, SCAN FOR SLOT
             52    08   D1     0BB6  1242          CMPL    #NBRK,R2                ;CHECK FOR LEGAL
                 F2    19     0BB9  1243          BLSS    15$                     ;OUT OF RANGE
           F809 CF42   D4     0BBB  1244 30$:      CLRL    BRKDSP[R2]              ;CLEAR DISPLAY
           F824 CF42   D4     0BC0  1245          CLRL    BRKCOM[R2]              ;CLEAR COMMAND ADDRESS
          50    D8 AB   D0     0BC5  1246          MOVL    F1-B(R11),R0            ;GET BREAKPOINT ADDRESS
                 16    13     0BC9  1247          BEQL    35$                     ;ALLOW CLEAR OF BREAKPOINT
                               0BCB  1248          .IF     DF,SW_PROCESS
                 3F    BB     0BCB  1249          PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ;SAVE REGISTERS FOR PROTECTION CHANGE
             54    50   D0     0BCD  1250          MOVL    R0,R4                   ;SET START ADDRESS
             55    50   D0     0BD0  1251          MOVL    R0,R5                   ;AND END ADDRESS
                07ED    30     0BD3  1252          BSBW    SETWRT                  ;SET PAGE WRITABLE
             50    6E   D0     0BD6  1253          MOVL    (SP),R0                 ;RESTORE BPT ADDRESS
                               0BD9  1254          .ENDC
             60    60   90     0BD9  1255          MOVB    (R0),(R0)               ;TEST WRITABILITY OF ADDRESS
                               0BDC  1256          .IF     DF,SW_PROCESS
                081B    30     0BDC  1257          BSBW    REPROT                  ;RESTORE PROTECTION
                  3F    BA     0BDF  1258          POPR    #^M<R0,R1,R2,R3,R4,R5>  ;AND REGISTERS
                               0BE1  1259          .ENDC
          0C 6A    0A   E1     0BE1  1260 35$:      BBC     #V_F3,(R10),40$         ;DISPLAY SPECIFIED?
      F7DD CF42    E0 AB   D0     0BE5  1261          MOVL    F3=B(R11),BRKDSP[R2]    ;SET DISPLAY START
                 03    13     0BEC  1262          BEQL    40$                     ;SKIP TEST IF NULL
              E0    BB   D5     0BEE  1263          TSTL    @F3-B(R11)              ;CHECK READABILITY
          07 6A    08   E1     0BF1  1264 40$:      BBC     #V_F4,(R10),45$         ;SKIP IF NO COMMAND ADDRESS
      F7ED CF42    E4 AB   D0     0BF5  1265          MOVL    F4=B(R11),BRKCOM[R2]    ;SET COMMAND STRING
         F79A CF42    50   D0     0BFC  1266 45$:      MOVL    R0,BRKADR[R2]           ;SAVE BREAKPOINT ADDRESS
                FEC7    31     0C02  1267          BRW     RESET                   ;RESET SCANNER AND RETURN
                               0C05  1268 ;
                               0C05  1269 ;       SHOBRK
                               0C05  1270 ;
                               0C05  1271 $HOBRK:
             55    01   D0     0C05  1272          MOVL    #1,R5                   ;INIT INDEX FOR LOOP
       58    F78F CF45   D0     0C08  1273 10$:      MOVL    BRKADR[R5],R8           ;GET BREAKPOINT ADDRESS
                 2E    13     0C0E  1274          BEQL    20$                     ;SKIP IF NULL
             53    55   D0     0C10  1275          MOVL    R5,R3                   ;BREAKPOINT NUMBER
                FD78    30     0C13  1276          BSBW    CRLF                    ;NEW LINE
                FCAB    30     0C16  1277          BSBW    OUTDIGIT                ;BPT NUMBER
                FD70    30     0C19  1278          BSBW    OUTSPACE                ;SPACE
             53    58   D0     0C1C  1279          MOVL    R8,R3                   ;ADDRESS OF BPT
                FCA6    30     0C1F  1280          BSBW    OUTLONG                 ;OUTPUT ADDRESS
                FD67    30     0C22  1281          BSBW    OUTSPACE                ;SPACE OVER
       53    F79F CF45   D0     0C25  1282          MOVL    BRKDSP[R5],R3           ;GET DISPLAY START
                 03    13     0C2B  1283          BEQL    15$                     ;NONE
                FC98    30     0C2D  1284          BSBW    OUTLONG                 ;OUTPUT DISPLAY START
```

```
                53   F7B4 CF45    D0  0C30  1285 15$:    MOVL    BRKCOM[R5],R3           ; GET COMMAND STRING ADDRESS
                               06    13  0C33  1286          BEQL    20$                     ; NONE
                             FD51    30  0C35  1287          BSBW    OUTSPACE                ; SPACE ANOTHER
                             FC8A    30  0C38  1288          BSBW    OUTLONG                 ; AND OUTPUT A LONGWORD
              FFC4 55    01    08    F1  0C3B  1289 20$:     ACBL    #NBRK,#1,R5,10$         ; DO THEM ALL
                             FD4A    31  0C41  1290          BRW     CRLF                    ; AND EXIT THROUGH CRLF
```

```
                                0C47  1292          .SBTTL  GO - START EXECUTION AT SPECIFIED LOCATION
                                0C47  1293 ;
                                0C47  1294 ;+       GO
                                0C47  1295 ;
      05 6A    08   E1          0C47  1296 GO:      BBC     #V_F1,(R10),PROCED      ; JUST PROCEED IF NO VALUE
   54 AB    DB AB   D0          0C4B  1297          MOVL    F1=B(R11),SAVPC-B(R11)  ; SET NEW PC
                                0C50  1298 ;        BRW     PROCED                  ; FALL INTO PROCEED
                                0C50  1299 ;
                                0C50  1300 ;+       PROCEED
                                0C50  1301 ;
                                0C50  1302 PROCED:                                  ;
6A   80008000 8F    CA          0C50  1303          BICL    #<<1@V_PRMODE>!<1@V_PREG>>,(R10)        ; CLEAR PROCESSOR REGISTER M
                    04          0C57  1304          RET                             ; RETURN
```

```
                        0C58  1306          .SBTTL  SEMI-I, PC VALUE
                        0C58  1307 ;
                        0C58  1308 ;       SEMI-I
                        0C58  1309 ;
          F980    30    0C58  1310 COLON:  BSBW    ENDEXPR                 ; TERMINATE EXPRESSION
     F0 AB  57    D0    0C5B  1311         MOVL    R7,PID-B(R11)           ; SET PID FOR PROCESS
            56    7C    0C5F  1312         CLRQ    R6                      ; RESET ACCUMULATORS
                  05    0C61  1313         RSB                             ;
                        0C62  1314
     51   EC AB   DE    0C62  1315 MFYFLGS:MOVAL   MFYFLG-B(R11),R1        ; SET MODIFY FLAG ADDRESS
                  17    0C66  1316         BRB     VALUE                   ; SET/GET VALUE
        51   6B   DE    0C68  1317 DOT:    MOVAL   CURDOT-B(R11),R1        ; SET ADDRESS OF DOT
     18 6A   1F    E1    0C6B  1318         BBC     #V_PREG,(R10),VALR      ; WAS IT PROCESSOR REGISTER?
     14 6A   0F    E2    0C6F  1319         BBSS    #V_PRMODE,(R10),VALR    ; YES, SET PROCESSOR REGISTER MODE
                  12    0C73  1320         BRB     VALR                    ; READ VALUE
     51   04 AB   DE    0C75  1321 QUANT:  MOVAL   QUAN-B(R11),R1          ; SET QUANTITY ADDRESS
                  0C    0C79  1322         BRB     VALR                    ; READ VALUE
                        0C7B  1323 PROGCTR:                                ;
     51   54 AB   DE    0C7B  1324         MOVAL   SAVPC-B(R11),R1         ; SET PC ADDRESS
     04 6A   08    E1    0C7F  1325 VALUE:  BBC     #V_F1,(R10),VALR        ; SKIP IF NO VALUE
     61   D8 AB   D0    0C83  1326         MOVL    F1-B(R11),(R1)          ; SET NEW VALUE FOR PC
        56   61    D0    0C87  1327 VALR:   MOVL    (R1),R6 ; AND GET VALUE
             F935  31    0C8A  1328 VALI:   BRW     INFLD                   ; SET FIELD IN PROGRESS
                        0C8D  1329 REGISTER:                               ;
     55   18 AB   DE    0C8D  1330         MOVAL   SAVREG-B(R11),R5        ; SET BASE OF REGISTER AREA
                  02    0C91  1331         BSBB    REGCOM                  ; FETCH ADDRESS
                  F5    0C93  1332         BRB     VALI                    ; AND USE IT
             FD03 30    0C95  1333 REGCOM: BSBW    GETCHAR                 ; GET SECOND CHAR
F87B CF   10   58   3A    0C98  1334         LOCC    R8,#16,PRIMARY          ; TRANSLATE TO HEX
                        0C9E  1335         .IF     DF,SW_PROCESS           ; FOR PROCESS VERSION
                  21    12    0C9E  1336         BNEQ    10$                 ; LEGAL HEX DIGIT
  FE A9   4958 8F    B1    0CA0  1337         CMPW    #^A/XI/,-2(R9)          ; CHECK FOR EXIT COMMAND
                  40    12    0CA6  1338         BNEQ    ERR3                ; NO, ERROR
                60 AB   D5    0CA8  1339         TSTL    ASTEN-B(R11)            ; WERE ASTS ENABLED ON DELTA ENTRY?
                  09    13    0CAB  1340         BEQL    5$                  ; IF EQL NO, DON'T REENABLE THEM HERE
                        0CAD  1341         $SETAST_S       #1              ; YES, UNDO AST DISABLE BY DELTA
                        0CB6  1342 5$:     $EXIT_S EXITCODE                ; EXIT
                        0CC1  1343         .IFF                            ;
                        0CC1  1344         BEQL    ERR3                    ; ERROR, NOT HEX
                        0CC1  1345         .ENDC                           ;
                        0CC1  1346 10$:                                    ;
        50   10   50    C3    0CC1  1347         SUBL3   R0,#16,R0           ; INVERT
           56   6540    DE    0CC5  1348         MOVAL   (R5)[R0],R6         ; ACCUMULATE
                  05    0CC9  1349         RSB                             ; RETURN
                        0CCA  1350
        1A 6A   09    E1    0CCA  1351 XSET:   BBC     #V_F2,(R10),ERR3        ; ERROR IF NOT TWO FIELDS
51   DC AB   04    00    EF    0CCE  1352         EXTZV   #0,#4,F2-B(R11),R1      ; GET REGISTER NUMBER
     51   F734 CF41    DE    0CD4  1353         MOVAL   XREGV[R1],R1            ; AND COMPUTE REGISTER ADDRESS
                  A3    11    0CDA  1354         BRB     VALUE                   ; PROCESS VALUE
                        0CDC  1355 XREG:                                   ; X-REGISTER VALUE
        55   F72D CF    DE    0CDC  1356         MOVAL   XREGV,R5            ; SET ADDRESS OF REGISTER VECTOR
                  B2    10    0CE1  1357         BSBB    REGCOM              ; ADDRESS TO R6
               56   66    D0    0CE3  1358         MOVL    (R6),R6             ; GET VALUE
                  A2    11    0CE6  1359         BRB     VALI                ; AND NOTE INPUT IN FIELD
                        0CE8  1360         .ALIGN  LONG                    ; LONGWORD ALIGN EXCEPTION ROUTINES
                        0CE8  1361 XDELACV:                                ; ACCESS VIOLATION HANDLER
                        0CE8  1362 MCHK:                                   ; MACHINE CHECK
```

```
                OCE8  1363          .IF     NDF,SW_PROCESS                  ;
                OCE8  1364          TSTL    AP                              ; CHECK FOR SIMULATOR
                OCE8  1365          BNEQ    ERR3                            ; YES, SKIP RESET
                OCE8  1366                                                  ;
                OCE8  1367                                                  ;
                OCE8  1368          CPUDISP <<780,CLR_780>,-                ; *DISPATCH ON CPU TYPE*
                OCE8  1369                   <750,CLR_750>,-                ;
                OCE8  1370                   <730,CLR_730>,-
                OCE8  1371                   <790,CLR_790>,-
                OCE8  1372                   <UV1,CLR_UV1>>,-
                OCE8  1373                  ENVIRON=XDELTA;
                OCE8  1374                                                  ;
                OCE8  1375 CLR_780:                                         ; FOR 11/780:
                OCE8  1376          MFPR    #PR$_SBIFS,-(SP)                ; GET ERROR BITS
                OCE8  1377          BBCC    #25,(SP),10$                    ; CLEAR ERROR 1ST PASS BIT
                OCE8  1378 10$:     MTPR    (SP)+,#PR$_SBIFS                ; CLEAR SBI FAULT
                OCE8  1379          BRB     CLR_END                         ; ERROR CLEARED
                OCE8  1380                                                  ;
                OCE8  1381 CLR_UV1:                                         ; FOR MicroVAX I:
                OCE8  1382 CLR_730:                                         ; FOR 11/730:
                OCE8  1383 CLR_750:                                         ; FOR 11/750:
                OCE8  1384          MTPR    #^XF,#PR$_MCESR                 ; SET 1 TO CLEAR MCHECK ERROR SUMMARY
                OCE8  1385          BRB     CLR_END                         ;
                OCE8  1386 CLR_790:
                OCE8  1387          MFPR    #PR$_EHSR,-(SP)                 ; GET ERR HANDLING STATUS REGISTER
                OCE8  1388          BBCC    #6,(SP),10$                     ; CLEAR VMS ENTERED BIT
                OCE8  1389 10$:     MTPR    (SP)+,#PR$_EHSR                 ; WRITE BACK TO CLEAR
                OCE8  1390          JSB     SYSL$CLRSBIA                    ; CLEAR SBIA ERROR BITS
                OCE8  1391                                                  ;
                OCE8  1392 CLR_END:                                         ; *END OF CPU-DEPENDENT CODE*
                OCE8  1393                                                  ;
                OCE8  1394                                                  ;
                OCE8  1395                  .ENDC                           ;
                OCE8  1396 10$:                                             ;
    F86B    31  OCE8  1397 ERR3:    BRW     ERROR                           ; AND DECLARE ERROR
                OCEB  1398
```

```
                        OCEB  1400          .SBTTL  REGISTER SAVE AND RESTORE
                        OCEB  1401
                        OCEB  1402 ;
                        OCEB  1403 ;       SAVE - SAVE TARGET REGISTERS, PC, PSL
                        OCEB  1404 ;
                        OCEB  1405 $SAVE:
                        OCEB  1406          .IF     NDF,SW_PROCESS
                        OCEB  1407          SETIPL  #31                     ; DISABLE
                        OCEB  1408          JSB     INI$WRITABLE            ; MAKE THE SYSTEM WRITABLE
                        OCEB  1409          MOVQ    R0,SAVREG               ; SAVE R0,R1
                        OCEB  1410          MOVAB   SAVR2,R1                ; SETUP BASE FOR REMAINING REGS
                        OCEB  1411          .IFF                            ; FALSE IF PROCESS VERSION
                        OCEB  1412          $SETAST_S       #0              ; DISABLE ASTS
                  70    9F    OCF4  1413          PUSHAB  -(R0)               ; SAVE ENABLE VALUE-1
                  51    DC    OCF6  1414          MOVPSL  R1                  ; GET CURRENT PSL
          51  51   02    18    EF    OCF8  1415          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R1,R1       ; ISOLATE CURRENT MODE
             51   00E4  8F    A4    OCFD  1416          MULW    #CONTEXTSZ,R1       ; COMPUTE OFFSET TO PROPER CONTEXT AREA
          51   F39D  CF41    9E    0D02  1417          MOVAB   SAVREG[R1],R1       ; FORM ADDRESS OF REGISTER SAVE
             50    08  AC    D0    0D08  1418          MOVL    8(AP),R0            ; GET POINTER TO MECHANISM
             81    0C  A0    7D    0D0C  1419          MOVQ    12(R0),(R1)+        ; SAVE R0,R1
                        0D10  1420          .ENDC
                  81    52    7D    0D10  1421          MOVQ    R2,(R1)+            ; SAVE R2,R3
                  81    54    7D    0D13  1422          MOVQ    R4,(R1)+            ; SAVE R4,R5
                  81    56    7D    0D16  1423          MOVQ    R6,(R1)+            ; SAVE R6,R7
                  81    58    7D    0D19  1424          MOVQ    R8,(R1)+            ; SAVE R8,R9
                  81    5A    7D    0D1C  1425          MOVQ    R10,(R1)+           ; SAVE R10,R11
                        0D1F  1426          .IF     NDF,SW_PROCESS
                        0D1F  1427          MOVQ    AP,(R1)+                ; SAVE AP,FP
                        0D1F  1428          MOVAB   12(SP),(R1)+            ; ASSUME KERNEL STACK
                        0D1F  1429          MOVQ    4(SP),(R1)+             ; SAVE PC,PSL
                        0D1F  1430          .IFF
                     81 08  AD    7D    0D1F  1431          MOVQ    8(FP),(R1)+         ; SAVE AP,FP
                  50 04  BC    01    C3    0D23  1432          SUBL3   #1,@4(AP),R0        ; GET NUMBER OF ARGS IN SIGNAL
                     50  04  BC40    DE    0D28  1433          MOVAL   @4(AP)[R0],R0       ; POINT TO PC,PSL
                     81  08  A0      DE    0D2D  1434          MOVAL   8(R0),(R1)+         ; COMPUTE SP
                         81  60      7D    0D31  1435          MOVQ    (R0),(R1)+          ; SAVE PC,PSL
                        0D34  1436          .ENDC
                        0D34  1437          .IF     NDF,SW_PROCESS
                        0D34  1438          MOVL    R1,R2                   ; SAVE R1
                        0D34  1439          JSB     G^CON$OWNCTY            ; ALLOCATE THE CONSOLE TERMINAL
                        0D34  1440          MOVL    R0,(R2)+                ; SAVE CONSOLE TRANSMIT STATUS
                        0D34  1441          MOVL    R1,(R2)+                ; SAVE CONSOLE RECVR STATUS
                        0D34  1442          MOVL    R2,R1                   ; RESTORE R1
                        0D34  1443          CLRL    AP                      ; ZAP DEVICE ADDRESS BASE
                        0D34  1444 20$:     MOVAB   B,R11                   ; AND DATA BASE ADDRESS
                        0D34  1445          .IFF                            ; FALSE FOR PROCESS VERSION
                  5B    FFA4  C1    9E    0D34  1446          MOVAB   W^<B-<SAVPSL+4>>(R1),R11
                                                                            ; SET BASE OF CONTEXT AREA
                     60 AB      8E    D0    0D39  1447          MOVL    (SP)+,ASTEN-B(R11)  ; SAVE AST ENABLE
                        0D3D  1448          .ENDC
                     5A  D4  AB    9E    0D3D  1449          MOVAB   STATUS-B(R11),R10   ; SET STATUS BASE
                     59  84  AB    9E    0D41  1450          MOVAB   INBUF-B(R11),R9     ; POINT TO INPUT BUFFER
                                 69    94    0D45  1451          CLRB    (R9)                ; MAKE BUFFER EMPTY
                        0D47  1452          .IF     NDF,SW_PROCESS
                        0D47  1453          BSBW    GET$CB                  ; GET BASE OF SCB
                        0D47  1454          MOVL    4(R0),MCHKSAV           ; SAVE ORIGINAL MCHK VECTOR
                        0D47  1455          MOVAB   MCHK,4(R0)              ; SET TO XDELTA VECTOR
                        0D47  1456          MOVAB   XDELACV,^X20(R0)        ; SET ACCESS VIOLATION VECTOR
```

```
                        0D47  1457          MOVAB   XDELACV,^X24(R0)          ; SET PG FAULT VECTOR
                        0D47  1458          MOVAB   XDELACV,^X18(R0)          ; SET RESERVED OPERAND HANDLER
                        0D47  1459          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,8(SP),R0     ; GET MODE
                        0D47  1460          BEQL    30$                       ; CORRECT ALREADY IF KERNEL
                        0D47  1461          ADDL    #PR$_KSP,R0               ; COMPUTE PROCESSOR REGISTER
                        0D47  1462          MFPR    R0,SAVSP-B(R11)           ; AND SAVE CORRECT SP
                        0D47  1463          .ENDC
                FD82  31  0D47  1464  30$:  BRW     RESET                     ; RESET SCANNER
                        0D4A  1465
                        0D4A  1466  ;
                        0D4A  1467  ;       RESTORE - RESTORE TARGET REGISTERS
                        0D4A  1468  ;
                        0D4A  1469  RESTORE:                                  ; RESTORE EVERYTHING
                        0D4A  1470          .IF     NDF,SW_PROCESS
                        0D4A  1471          MOVQ    SAVPC-B(R11),4(SP)        ; SET PC,PSL
                        0D4A  1472          .IFF                              ; FALSE IF PROCESS
          50  04 BC  01  C3  0D4A  1473          SUBL3   #1,@4(AP),R0          ; GET SIGNAL ARG COUNT
            50  04 BC40  DE  0D4F  1474          MOVAL   @4(AP)[R0],R0         ; COMPUTE ADDRESS OF PC,PSL
              60  54 AB  7D  0D54  1475          MOVQ    SAVPC-B(R11),(R0)     ; STORE UPDATED PC,PSL
                        0D58  1476          .ENDC
                        0D58  1477  RESTORR:                                  ; RESTORE REGISTERS ONLY
                        0D58  1478          .IF     NDF,SW_PROCESS
                        0D58  1479          BSBB    GET$CB                    ; GET BASE OF SCB
                        0D58  1480          MOVAB   EXE$ACVIOLAT,^X20(R0)     ; RESTORE ACCESS VECTOR
                        0D58  1481          MOVAB   MMG$PAGEFAULT,^X24(R0)    ; AND PAGE FAULT VECTOR
                        0D58  1482          MOVL    MCHKSAV,4(R0)             ; RESTORE MACHINE CHECK VECTOR
                        0D58  1483          MOVAB   EXE$ROPRAND,^X18(R0)      ; RESTORE RESERVED OPERAND VECTOR
                        0D58  1484          TSTW    AP                        ; CHECK FOR CONSOLE
                        0D58  1485          BNEQ    10$                       ; NO, OTHER DEVICE
                        0D58  1486          MOVL    SAVOCR-B(R11),R0          ; RESTORE INITIAL TX STATUS
                        0D58  1487          MOVL    SAVRXCS-B(R11),R1         ; AND INITIAL RECEIVER STATE
                        0D58  1488          JSB     G^CON$RELEASECTY          ;
                        0D58  1489          BRB     20$                       ; MERGE WITH COMMON CODE
                        0D58  1490
                        0D58  1491  10$:    MOVW    SAVOCR-B(R11),OUTCR(AP)   ; RESTORE OUTPUT CSR
                        0D58  1492          MOVW    SAVRCR-B(R11),RDCR(AP)    ; AND INPUT CSR CONTENT
                        0D58  1493          .IFF
              60  AB  DD  0D58  1494          PUSHL   ASTEN-B(R11)              ; SAVE AST ENABLE
                        0D5B  1495          .ENDC
          51  20  AB  9E  0D5B  1496  20$:    MOVAB   SAVR2-B(R11),R1           ; SET BASE FOR RESTORE
            52  81  7D  0D5F  1497          MOVQ    (R1)+,R2                  ; RESTORE R2,R3
            54  81  7D  0D62  1498          MOVQ    (R1)+,R4                  ; RESTORE R4,R5
            56  81  7D  0D65  1499          MOVQ    (R1)+,R6                  ; RESTORE R6,R7
            58  81  7D  0D68  1500          MOVQ    (R1)+,R8                  ; RESTORE R8,R9
            5A  81  7D  0D6B  1501          MOVQ    (R1)+,R10                 ; RESTORE R10,R11
                        0D6E  1502          .IF     NDF,SW_PROCESS
                        0D6E  1503          MOVQ    (R1)+,AP                  ; RESTORE AP,FP
                        0D6E  1504          MOVQ    SAVREG,R0                 ; RESTORE R0,R1
                        0D6E  1505          .IFF                              ; FALSE IF PROCESS VERSION
          08  AD  81  7D  0D6E  1506          MOVQ    (R1)+,8(FP)               ; SET NEW VALUES FOR AP,FP
            50  08  AC  D0  0D72  1507          MOVL    8(AP),R0                  ; GET MECHANISM POINTER
        0C  A0  C8  A1  7D  0D76  1508          MOVQ    <SAVREG-SAVSP>(R1),12(R0)         ; STORE UPDATED R0,R1
                51  DC  0D7B  1509          MOVPSL  R1                        ; GET CURRENT PSL
    51  51  02  18  EF  0D7D  1510          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R1,R1         ; GET CURRENT MODE
      00  F74D  CF  51  E5  0D82  1511          BBCC    R1,DBGACTIVE,30$          ; CLEAR ACTIVE BIT FOR MODE
                        0D88  1512  30$:
                    BE  D5  0D88  1513          TSTL    (SP)+                     ; CHECK FOR AST ENABLE
```

```
                09   13  0D8A  1514          BEQL    35$                     ; NO
                         0D8C  1515          $SETAST_S       #1              ; RE- ENABLE AST RECOGNITION
                         0D95  1516 35$:
                         0D95  1517          .ENDC
                         0D95  1518          .IF     NDF,SW_PROCESS
                         0D95  1519          JSB     INI$RDONLY              ; REPROTECT THE SYSTEM CODE
                         0D95  1520          .ENDC
                    05   0D95  1521          RSB                             ; AND RETURN
```

```
                        .SBTTL  GET SCB ADDRESS

                ;
                ; SUBROUTINE GETSCB IS CALLED TO GET THE PHYSICAL OR VIRTUAL
                ; ADDRESS OF THE CURRENT SCB.
                ;
                ; INPUTS:       NONE
                ;
                ; OUTPUTS:      R0 = SCB ADDRESS
                ;               OTHER REGISTERS PRESERVED
                ;

                        .IF     NDF,SW_PROCESS          ; NOT FOR PROCESS VERSION
                GETSCB: MFPR    #PR$_MAPEN,R0           ; GET MAPPING STATUS
                        BNEQ    10$                     ; BRANCH IF MAPPING ENABLED
                        MFPR    #PR$_SCBB,R0            ; ELSE GET PHY ADDR OF SCB
                        BRB     20$                     ; JOIN COMMON RETURN
                10$:    MOVL    EXE$GL_SCB,R0           ; IF MAPPING ENABLED, GET SCB VA
                20$:    RSB                             ; RETURN
                        .ENDC                           ;
```

```
                                                 0D96  1545          .SBTTL  BPT TRAP HANDLER
                                                 0D96  1546 ;
                                                 0D96  1547 ;       HANDLE BREAKPOINT TRAPS
                                                 0D96  1548 ;
00 20 54 41 20 4B 52 42 20                       0D96  1549 BMSG:    .ASCIZ  / BRK AT /                  ; BREAK POINT MESSAGE
                                                 0D9F  1550          .ALIGN  LONG                        ; LONGWORD ALIGNMENT
                                                 0DA0  1551          .IF     NDF,SW_PROCESS              ; EXEC VERSION
                                                 0DA0  1552 XDELBPT::                                    ; XDELTA BPT ENTRY
                                                 0DA0  1553          .IFF                                ;
                                                 0DA0  1554 XDELBPT:                                     ; DELTA BPT ENTRY
                                                 0DA0  1555          .ENDC                               ;
                                      FF48   30  0DA0  1556          BSBW    SAVE                        ; SAVE REGS AND DISABLE
                                      0119   30  0DA3  1557          BSBW    GETBPTX                     ; GET INDEX OF BPT
                                        53   D5  0DA6  1558          TSTL    R3                          ; CHECK FOR MATCH
                                        17   12  0DA8  1559          BNEQ    10$                         ; YES, FOUND IT
                                F610 CF 54 AB   D1  0DAA  1560          CMPL    SAVPC-B(R11),OVRADR         ; IS THIS A TEMPORARY BREAKPOINT?
                                        06   13  0DB0  1561          BEQL    20$                         ; BRANCH IF SO
                                      FFA3   30  0DB2  1562          BSBW    RESTORR                     ; RESTORE REGISTERS ONLY
                                                 0DB5  1563          .IF     NDF,SW_PROCESS              ;
                                                 0DB5  1564          MOVZBL  6(SP),-(SP)                 ; GET IPL
                                                 0DB5  1565          ENBINT                              ; ENABLE
                                                 0DB5  1566          JMP     EXE$BREAK                   ; AND HANDLE NORMALLY
                                                 0DB5  1567          .IFF                                ; FALSE IF PROCESS VERSION
                                                 0DB5  1568 ;
                                                 0DB5  1569 ;       ***** UNEXPECTED BREAKPOINT *****
                                           50    D4  0DB5  1570          CLRL    R0                          ; RETURN FALSE
                                                 04  0DB7  1571          RET                                 ;
                                                 0DB8  1572          .ENDC                               ;
                                                 0DB8  1573 ;
                                                 0DB8  1574 ; WE JUST HIT A TEMPORARY BREAKPOINT SET FROM A STEP-OVER
                                                 0DB8  1575 ;
                                        00A6   30  0DB8  1576 20$:     BSBW    UNBRK                       ; RESTORE OPCODES, INCLUDING TEMP BRKPT
                                    F601 CF    D4  0DBB  1577          CLRL    OVRADR                      ; REMOVE TEMPORARY BREAKPOINT
                                          40   11  0DBF  1578          BRB     OUTPC                       ; AND PRETEND WE JUST STEPPED
                                                 0DC1  1579 ;
                                    6A    18   88  0DC1  1580 10$:     BISB    #<<1@V_TBIT>!<1@V_ATBRK>>,(R10) ; SET STATUS
                                                 0DC4  1581 30$:                                         ;
                                        009A   30  0DC4  1582          BSBW    UNBRK                       ; RESTORE OPCODES
                                 4D 58 AB   04   E0  0DC7  1583          BBS     #PSL$V_TBIT,SAVPSL-B(R11),PROCEED   ; PROCEED IF BPT AND TBIT
                                    55      53   D0  0DCC  1584          MOVL    R3,R5                       ; SAVE BPT NUMBER
                                          FBBF   30  0DCF  1585          BSBW    CRLF                        ; OUTPUT CR/LF PAIR
                                          FAEF   30  0DD2  1586          BSBW    OUTDIGIT                    ; OUTPUT BPT NUMBER
                                 54       BE AF   9E  0DD5  1587          MOVAB   BMSG,R4                     ; MSG ADDRESS
                                          FB09   30  0DD9  1588          BSBW    OUTZSTRING                  ; OUTPUT ASCIIZ
                                 53       54 AB   D0  0DDC  1589          MOVL    SAVPC-B(R11),R3             ; OUTPUT PC
                                          FAE5   30  0DE0  1590          BSBW    OUTLONG                     ; OUTPUT HEX LONGWORD
                              51     F5E1 CF45   D0  0DE3  1591          MOVL    BRKDSP[R5],R1               ; GET ADDRESS TO DISPLAY
                                            08   13  0DE9  1592          BEQL    40$                         ; NONE
                                    6B      51   D0  0DEB  1593          MOVL    R1,CURDOT-B(R11)            ; SET AS CURRENT DOT
                                  6A    2000 8F   AA  0DEE  1594          BICW    #1@V_INSTR,(R10)            ; CLEAR INSTRUCTION DISPLAY MODE
                                          F8F4   30  0DF3  1595          BSBW    LOCPROMPT                   ; AND DISPLAY
                              51     F5EE CF45   D0  0DF6  1596 40$:     MOVL    BRKCOM[R5],R1               ; GET COMMAND STRING ADDRESS
                                            03   13  0DFC  1597          BEQL    OUTPC                       ; NONE OUTPUT INSTRUCTION AT PC
                                    59      51   D0  0DFE  1598          MOVL    R1,R9                       ; SET TO SCAN STORED COMMAND
                                                 0E01  1599 OUTPC:                                       ; OUTPUT PC INSTRUCTION & GET COMMANDS
                                    6B   54 AB   D0  0E01  1600          MOVL    SAVPC-B(R11),CURDOT-B(R11)      ; SET ADDRESS
                                                 0E05  1601          IFNORD  #4,@CURDOT-B(R11),GETCMD ; SKIP DISPLAY IF NOT READABLE
```

```
                6A   2000 8F    AB  0E0C  1602          BISW    #1@V_INSTR,(R10)        ; SET TO INSTRUCTION DISPLAY MODE
                       F8D6    30  0E11  1603          BSBW    LOCPROMPT               ; PROMPT WITH ADDRESS/INSTRUCTION
                               0E14  1604 GETCMD:                                      ; GET COMMANDS
                F734 CF    6C   FA  0E14  1605          CALLG   (AP),DCOM               ; PERFORM DEBUG COMMANDS
                               0E19  1606 PROCEED:                                     ; PROCEED
                           6E  10  0E19  1607          BSBB    SETBRK                  ; SET BREAKPOINTS
                     20 6A     03  E5  0E1B  1608          BBCC    #V_TBIT,(R10),50$       ; TEST AND CLR TRACE FLAG
                00 58 AB     04  E2  0E1F  1609 30$:      BBSS    #PSL$V_TBIT,SAVPSL-B(R11),40$   ; SET TBIT
                               0E24  1610 40$:                                         ;
                               0E24  1611          .IF     DF,SW_PROCESS           ; FOR PROCESS VERSION
                     54 BB     02  91  0E24  1612          CMPB    #2,@SAVPC-B(R11)        ; CHECK FOR REI OPCODE
                               11  12  0E28  1613          BNEQ    45$                     ; NO, NOTHING SPECIAL
          50    58 AB     02    18  EF  0E2A  1614          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,SAVPSL-B(R11),R0    ; GET NEW MODE
                50    00E4 8F     A4  0E30  1615          MULW    #CONTEXTSZ,R0           ; SCALE BY PER MODE CONTEXT AREA SIZE
                5A   F226 CF40    9E  0E35  1616          MOVAB   STATUS[R0],R10  ; POINT TO NEW FLAGS
                               0E3B  1617          .ENDC                           ;
                00 6A     05    E2  0E3B  1618 45$:      BBSS    #V_TBITOK,(R10),50$      ; SET TBIT EXPECTED
                       FF08    30  0E3F  1619 50$:      BSBW    RESTORE                 ; RESTORE EVERYTHING
                               0E42  1620          .IF     NDF,SW_PROCESS          ;
                               0E42  1621          REI                             ; AND RETURN
                               0E42  1622          .IFF                            ; FALSE IF PROCESS VERSION
                     50   01    D0  0E42  1623          MOVL    #1,R0                   ; RETURN TRUE
                               04  0E45  1624          RET                             ;
                               0E46  1625          .ENDC                           ;
                               0E46  1626
```

```
                     0E46  1628          .SBTTL  TBIT EXCEPTION HANDLER
                     0E46  1629 ;
                     0E46  1630 ;       HANDLER FOR TBIT EXCEPTION
                     0E46  1631 ;
                     0E46  1632          .ALIGN  LONG                            ; LONGWORD ALIGNED
                     0E48  1633          .IF     NDF,SW_PROCESS                  ;
                     0E48  1634 XDELTBIT::                                       ; XDELTA TBIT HANDLER
                     0E48  1635          .IFF                                    ;
                     0E48  1636 XDELTBIT:                                        ;
                     0E48  1637          .ENDC                                   ;
           FEA0   30 0E48  1638          BSBW    SAVE                            ; SAVE AND DISABLE
06 6A       05   E4 0E4B  1639          BBSC    #V_TBITOK,(R10),XDELDBG         ; BR IF TBIT EXPECTED
         FF06   30 0E4F  1640          BSBW    RESTORR                         ; RESTORE REGISTERS
                     0E52  1641          .IF     NDF,SW_PROCESS                  ;
                     0E52  1642          MOVZBL  6(SP),-(SP)                     ; GET IPL FOR ENABLE
                     0E52  1643          ENBINT                                  ; ENABLE
                     0E52  1644          JMP     EXE$TBIT                        ; OTHERWISE LET EXEC HANDLE
                     0E52  1645          .IFF                                    ; FALSE IF PROCESS VERSION
              50     D4 0E52  1646          CLRL    R0                          ; RESIGNAL
                 04   05 0E54  1647          RET                                 ; UNEXPECTED TBIT EXCEPTION
                     0E55  1648          .ENDC                                   ;
                     0E55  1649 XDELDBG:                                         ; COMMON WITH DEBUG EXCEPTION
58 AB       10   CA 0E55  1650          BICL    #<1@PSL$V_TBIT>,SAVPSL-B(R11)   ; CLEAR TBIT IN PSL
                 06   10 0E59  1651          BSBB    UNBRK                       ; REPLACE OPCODES
BA 6A       04   E4 0E5B  1652          BBSC    #V_ATBRK,(R10),PROCEED          ; CHECK FOR PROCEED
                 A0   11 0E5F  1653          BRB     OUTPC                       ; DISPLAY INSTRUCTION AND GET COMMANDS
                     0E61  1654
```

```
                                0E61  1656          .SBTTL  UNBRK - RESTORE OPCODES FOR BREAKPOINTS
                                0E61  1657 ;
                                0E61  1658 ;       UNBRK
                                0E61  1659 ;
                                0E61  1660 UNBRK:
                     51   09    D0  0E61  1661          MOVL    #NBRK+NTMPBRK,R1        ; TOTAL PERM & TEMPORARY BREAKPOINTS
                50   F533 CF41  D0  0E64  1662 10$:     MOVL    BRKADR[R1],R0           ; GET BREAKPOINT ADDRESS
                          19    13  0E6A  1663          BEQL    20$                     ; SKIP IF NOT ENABLED
                                0E6C  1664          .IF     DF,SW_PROCESS
                          3F    BB  0E6C  1665          PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; SAVE REGS FOR PROTECTION CHANGE
                       54   50    D0  0E6E  1666          MOVL    R0,R4                   ; FORM INADR RANGE FOR SET PROTECTION
                       55   50    D0  0E71  1667          MOVL    R0,R5
                          054C  30  0E74  1668          BSBW    SETWRT                  ; SET PAGE WRITABLE
                       50   6E    7D  0E77  1669          MOVQ    (SP),R0                 ; RESTORE R0,R1
                                0E7A  1670          .ENDC
                50   F544 CF41  90  0E7A  1671          MOVB    BRKOP[R1],(R0)          ; RESTORE OPCODE
                                0E80  1672          .IF     DF,SW_PROCESS
                          0577  30  0E80  1673          BSBW    REPROT                  ; RESTORE PROTECTION
                          3F    BA  0E83  1674          POPR    #^M<R0,R1,R2,R3,R4,R5>  ; RESTORE REGISTERS
                                0E85  1675          .ENDC
                          DC 51 F5  0E85  1676 20$:     SOBGTR  R1,10$                  ; DO THEM ALL
                                05  0E88  1677          RSB                             ; AND RETURN
                                0E89  1678
```

```
                        0E89  1680          .SBTTL  SETBRK - SET BREAK POINT INSTRUCTIONS
                        0E89  1681 ;
                        0E89  1682 ;       SETBRK
                        0E89  1683 ;
      51    09    D0    0E89  1684 SETBRK: MOVL    #NBRK+NTMPBRK,R1          ; TOTAL PERMANENT & TEMPORARY BRKPOINTS
50  F50B CF41   D0    0E8C  1685 10$:     MOVL    BRKADR[R1],R0             ; GET ADDRESS
        27    13    0E92  1686          BEQL    20$                       ; SKIP IF NOT ENABLED
F529 CF41   60    90    0E94  1687          MOVB    (R0),BRKOP[R1]            ; SAVE OPCODE
        6A    18    93    0E9A  1688          BITB    #<<1@V_TBIT>!<1@V_ATBRK>>,(R10) ; CHECK FOR TRACE
        06    13    0E9D  1689          BEQL    15$                       ; NO TRACE, SET ANYWAY
    54 AB     50    D1    0E9F  1690          CMPL    R0,SAVPC-B(R11)           ; CHECK FOR AT BPT
        16    13    0EA3  1691          BEQL    20$                       ; YES, DONT SET IT
                        0EA5  1692 15$:
                        0EA5  1693          .IF     DF,SW_PROCESS
        3F    BB    0EA5  1694          PUSHR   #^M<R0,R1,R2,R3,R4,R5>    ; SAVE REGISTERS FOR PROTECTION CHANGE
      54    50    D0    0EA7  1695          MOVL    R0,R4                     ; SET START ADDRESS OF RANGE
      55    50    D0    0EAA  1696          MOVL    R0,R5                     ; AND END ADDRESS
    0513    30    0EAD  1697          BSBW    SETWRT                    ;SET PAGE WRITABLE
      50    6E    D0    0EB0  1698          MOVL    (SP),R0                   ; RESTORE BPT ADDRESS
                        0EB3  1699          .ENDC
      60    03    90    0EB3  1700          MOVB    #3,(R0)                   ; SET BREAKPOINT OPCODE
                        0EB6  1701          .IF     DF,SW_PROCESS
    0541    30    0EB6  1702          BSBW    REPROT                    ; RESTORE ORIGINAL PROTECTION VALUE
        3F    BA    0EB9  1703          POPR    #^M<R0,R1,R2,R3,R4,R5>    ; AND REGISTERS
                        0EBB  1704          .ENDC
   CE 51    F5    0EBB  1705 20$:     SOBGTR  R1,10$                    ; DO THEM ALL
                05    0EBE  1706          RSB                               ; AND RETURN
                        0EBF  1707
```

```
                        OEBF  1709          .SBTTL  GETBPTX - GET INDEX FOR BREAKPOINT
                        OEBF  1710 ;
                        OEBF  1711 ;       GETBPTX
                        OEBF  1712 ;
                        OEBF  1713 GETBPTX:                                             ;
          53    08      DO    OEBF  1714            MOVL    #NBRK,R3                    ; INIT LOOP
F4D3 CF43     54  AB    D1    OEC2  1715 10$:       CMPL    SAVPC-B(R11),BRKADR[R3]     ; IS THIS A BPT?
                03  13    OEC9  1716            BEQL    20$                         ; YES
              F4 53   F5    OECB  1717            SOBGTR  R3,10$                      ; NO, CONTINUE
                      05    OECE  1718 20$:       RSB                                 ; RETURN
```

```
                     0ECF  1720          .SBTTL  QUOTE - INPUT CHARACTER STRING
                     0ECF  1721 ;
                     0ECF  1722 ;         QUOTE - START CHARACTER STRING INPUT
                     0ECF  1723 ;
                     0ECF  1724 QUOTE:                                        ;
55   6B   D0  0ECF  1725          MOVL    CURDOT-B(R11),R5                    ; POINT TO STRING BUFFER
   FAC6   30  0ED2  1726 5$:       BSBW    GETCHAR                             ; GET CHARACTER
58   27   91  0ED5  1727          CMPB    #QUOT,R8                            ; CHECK FOR QUOTE
     05   13  0ED8  1728          BEQL    10$                                 ; YES, END OF STRING
85   58   90  0EDA  1729          MOVB    R8,(R5)+                            ; INSERT IN BUFFER
     F3   11  0EDD  1730          BRB     5$                                  ; AND CONTINUE
6B   55   D0  0EDF  1731 10$:      MOVL    R5,CURDOT-B(R11)                    ; SAVE NEW DOT
          05  0EE2  1732          RSB                                         ; RETURN
```

```
                        OEE3  1734          .SBTTL  DEPOSIT
                        OEE3  1735 ;
                        OEE3  1736 ;       DEPOSIT DATA
                        OEE3  1737 ;
                        OEE3  1738 DEPOSIT:                                     ;
3F 6A   1F   E0         OEE3  1739          BBS     #V_PREG,(R10),40$           ; BR IF PROCESSOR REGISTER
                        OEE7  1740          .IF     DF,SW_PROCESS               ;
   54   6B   D0         OEE7  1741          MOVL    CURDOT-B(R11),R4            ; GET CURRENT DOT
     F0 AB   D5         OEEA  1742          TSTL    PID-B(R11)                  ; CHECK FOR ARBITRARY PROCESS DEPOSIT
        44   12         OEED  1743          BNEQ    50$                         ; BR IF YES
                        OEEF  1744          .ENDC                               ;
                        OEEF  1745          CASE    CURTYPE-B(R11),TYPE=B,<-    ;       ; SWITCH ON TYPE
                        OEEF  1746                  10$,-                       ; BYTE
                        OEEF  1747                  20$,-                       ; WORD
                        OEEF  1748                  30$,-                       ; LONG
                        OEEF  1749                  >                           ;
                        OEFA  1750          .IF     NDF,SW_PROCESS              ;
                        OEFA  1751 10$:     MOVB    F1-B(R11),@CURDOT-B(R11)    ; STORE BYTE
                        OEFA  1752          RSB                                 ; RETURN
                        OEFA  1753 20$:     MOVW    F1-B(R11),@CURDOT-B(R11)    ; STORE WORD
                        OEFA  1754          RSB                                 ; RETURN
                        OEFA  1755 30$:     MOVL    F1-B(R11),@CURDOT-B(R11)    ; STORE LONG
                        OEFA  1756          RSB                                 ; RETURN
                        OEFA  1757 40$:     MTPR    F1-B(R11),CURDOT-B(R11)     ; SET VALUE IN PROCESSOR REGISTER
                        OEFA  1758          RSB
                        OEFA  1759          .IFF                                ; FALSE IF PROCESS VERSION
                        OEFA  1760 10$:                                         ; BYTE DEPOSIT
   55   54   D0         OEFA  1761          MOVL    R4,R5                       ; START AND END ADDRESSES EQUAL
        04C3 30         OEFD  1762          BSBW    SETWRT                      ; SET WRITABLE, OLD PROT TO R2
64 D8 AB    90         OF00  1763          MOVB    F1-B(R11),(R4)              ; STORE BYTE
        04F3 30         OF04  1764          BSBW    REPROT                      ; RESTORE PROTECTION
             05         OF07  1765          RSB                                 ;
                        OF08  1766
55   54   01   C1       OF08  1767 20$:     ADDL3   #1,R4,R5                    ; WORD DEPOSIT, FORM END ADDRESS
        04B4 30         OF0C  1768          BSBW    SETWRT                      ; SET WRITABLE
64 D8 AB    B0         OF0F  1769          MOVW    F1-B(R11),(R4)              ; STORE WORD
        04E4 30         OF13  1770          BSBW    REPROT                      ; RESTORE PROTECTION
             05         OF16  1771          RSB                                 ;
                        OF17  1772
55   54   03   C1       OF17  1773 30$:     ADDL3   #3,R4,R5                    ; LONGWORD DEPOSIT, FORM END ADDRESS
        04A5 30         OF1B  1774          BSBW    SETWRT                      ; SET WRITABLE
64 D8 AB    D0         OF1E  1775          MOVL    F1-B(R11),(R4)              ; STORE LONG WORD
        04D5 30         OF22  1776          BSBW    REPROT                      ; RESTORE PROTECTION
             05         OF25  1777          RSB                                 ;
                        OF26  1778
                        OF26  1779 40$:                                         ; PROCESSOR REGISTER
                        OF26  1780          $CMKRNL_S       B^DEPPREG,(AP)      ; DEPOSIT IN PROCESSOR REGISTER
             05         OF32  1781          RSB                                 ;
                        OF33  1782 50$:                                         ; DEPOSIT IN ARBITRARY PROCESS
                        OF33  1783          CASE    CURTYPE-B(R11),TYPE=B,<-    ;       ; SWITCH ON TYPE
                        OF33  1784                  60$,-                       ; BYTE
                        OF33  1785                  70$,-                       ; WORD
                        OF33  1786                  80$>                        ; LONGWORD
             05         OF3E  1787          RSB                                 ;
 14EE'CF     9F         OF3F  1788 60$:     PUSHAB  W^DPBYTE                    ; SET ADDRESS OF BYTE ROUTINE
          0A 11         OF43  1789          BRB     90$                         ;
 1550'CF     9F         OF45  1790 70$:     PUSHAB  W^DPWORD                    ; SET ADDRESS OF WORD ROUTINE
```

```
                04   11   0F49  1791                BRB     90$                     ;
          15B1'CF    9F   0F4B  1792 80$:           PUSHAB  W^DPLONG                ; SET ADDRESS OF LONG ROUTINE
            F0 AB    DD   0F4F  1793 90$:           PUSHL   PID-B(R11)              ; SET PID OF TARGET
               6B    DD   0F52  1794                PUSHL   CURDOT-B(R11)           ; ADDRESS FOR STORE
            D8 AB    DD   0F54  1795                PUSHL   F1-B(R11)               ; VALUE TO STORE
               04    DD   0F57  1796                PUSHL   #4                      ; ARGUMENT COUNT
          50   5E    D0   0F59  1797                MOVL    SP,R0                   ; POINTER TO ARGUMENT LIST
            EC AB    D5   0F5C  1798                TSTL    MFYFLG-B(R11)           ; CHECK FOR STORE ENABLED
               0D    13   0F5F  1799                BEQL    100$                    ; BR IF NOT
                          0F61  1800                $CMKRNL_S       W^QGET,(R0)     ; CALL TO QUEUE REQUEST
          5E   14    C0   0F6E  1801 100$:          ADDL    #20,SP                  ; CLEAN STACK
                     05   0F71  1802                RSB                             ; AND RETURN
                          0F72  1803
                0000 0F72  1804 DEPPREG:.WORD      0                               ; DEPOSIT INTO PROCESSOR REGISTER
       6D 0F81'CF    9E   0F74  1805                MOVAB   W^PREXC,(FP)            ; SET EXCEPTION HANDLER
          6B   D8 AB    DA   0F79  1806                MTPR    F1-B(R11),CURDOT-B(R11) ; PLACE FIELD VALUE IN REG
               50   01    D0   0F7D  1807                MOVL    #1,R0                   ; RETURN SUCESS
                     04   0F80  1808                RET                             ;
                          0F81  1809
                0000 0F81  1810 PREXC:  .WORD      0                               ; PROCESSOR REGISTER EXCEPTION HANDLER
    51    08 AC    04   C1   0F83  1811                ADDL3   #4,8(AP),R1             ; POINT TO EXCEPTION FP
          0C AD    61   D0   0F88  1812                MOVL    (R1),12(FP)             ; SET AS RETURN FP
      10 AD   91'AF    9E   0F8C  1813                MOVAB   B^10$,16(FP)            ; SET RETURN ADDRESS
               50   01    3C   0F91  1814 10$:    MOVZWL  #1,R0           ;SET NORMAL STATUS
                     04   0F94  1815                RET                             ; AND RETURN
                          0F95  1816
                          0F95  1817                .ENDC
```

```
                        0F95  1819          .SBTTL  EXECUTE - PERFORM COMMAND STRING
                        0F95  1820 ;
                        0F95  1821 ;       EXECUTE
                        0F95  1822 ;
                        0F95  1823 EXECUTE:
09 6A    08     E1      0F95  1824          BBC     #V_F1,(R10),10$         ; EXIT IF NO ADDRESS
59   D8 AB     D0      0F99  1825          MOVL    F1=B(R11),R9            ; SET CHAR STRING
        03     12      0F9D  1826          BNEQ    10$                     ; NOT NULL
      F5BB     31      0F9F  1827          BRW     SUPERST                 ; SUPER RESET
               05      0FA2  1828 10$:     RSB                             ; RETURN
                        0FA3  1829
```

```
                        0FA3  1831          .SBTTL  P - PROCESSOR REGISTER PREFIX
                        0FA3  1832 ;
                        0FA3  1833 ;       SET PROCESSOR REGISTER MODE
                        0FA3  1834 ;
                        0FA3  1835 PREG:                                    ; PROCESSOR REGISTER MODE
00 6A   0F   E2         0FA3  1836          BBSS    #V_PRMODE,(R10),10$     ; SET PROCESSOR REG FLAG
             05         0FA7  1837 10$:     RSB                             ; RETURN
```

```
                                    0FA8  1839          .SBTTL  PROCESS DEBUGGER INITIALIZATION
                                    0FA8  1840
                                    0FA8  1841          .IF     DF,SW_PROCESS           ;
73 72 65 56 20 41 54 4C 45 44 0A 0D 0FA8  1842 SALUTE: .ASCIZ  <CR><LF>/DELTA Version X2.2/<CR><LF>    ;
   00 0A 0D 32 2E 32 58 20 6E 6F 69 0FB4
                                    0FBF  1843
                                    0FBF  1844 TEST:                                    ; START ADDRESS OF IMAGE ENTRY
                                    0FBF  1845 XDT$START::                              ; GLOBAL START ADDRESS FOR CLI DEBUG
                               0000 0FBF  1846          .WORD   0
                                    0FC1  1847 DELTA_START:                             ; START ADDRESS FOR DEBUGGER ENTRY
                                    0FC1  1848          $WAKE_S                         ; NULL WAKE AND
                                    0FC6  1849          $HIBER_S                        ; HIBERNATE TO GET SYNCHRONIZED
        F499 CF    F52D CF    9E    0FD3  1850          MOVAB   TERMASK,TERMASKADR      ; RELOCATE TERMINATOR MASK DESCR
        F47C CF    F483 CF    9E    0FDA  1851          MOVAB   TTNAMD+8,TTNAMD+4       ; RELOCATE DESCRIPTOR
        F497 CF    F49E CF    9E    0FE1  1852          MOVAB   DBGINPUT+8,DBGINPUT+4
        F4A1 CF    F4A8 CF    9E    0FE8  1853          MOVAB   TRNINPUT+8,TRNINPUT+4
     F4E4 CF   0000129C'EF    9E    0FEF  1854          MOVAB   EXIHANDLE,EXIHADR       ;
        F4E5 CF    F4EC CF    9E    0FF8  1855          MOVAB   EXITCODE,EXCODA         ; RELOCATE EXIT HANDLER ARGS
              0B'AF    6C    FA    0FFF  1856          CALLG   (AP),B^INITCALL         ; GENERATE CALL FRAME
                               04    1003  1857          RET                             ;
                                    1004  1858
              5C    04 AC    D0    1004  1859 NO0RK:   MOVL    4(AP),AP                ; GET EXCEPTION ARGUMENT LIST
                    01C4'    31    1008  1860          BRW     EXCEPT+2                ; AND GOTO EXCEPTION HANDLER
                                    1008  1861
                                    1008  1862 INITCALL:
                               0000 1008  1863          .WORD   0                       ; ENTRY MASK
           6D   1263'CF    9E    100D  1864          MOVAB   W^CATCHALL,(FP)         ; SET CATCHALL EXCEPTION HANDLER
                                    1012  1865          $CMKRNL_S       W^SETKEXC,(AP)  ; SET EXCEPTION VECTORS FROM KERNEL MODE
             16 50    E8    101F  1866          BLBS    R0,1$                           ; SUCCESS WITH KERNEL, IF NOT, TRY EXEC
                                    1022  1867          $CMEXEC_S       W^SETEEXC,(AP)  ; SET EXEC & SUPV EXCEPTION VECTORS
             06 50    E8    102F  1868          BLBS    R0,1$                           ; BRANCH IF SUCCESS
        24   50    D1    1032  1869          CMPL    R0,#SS$_NOPRIV                  ; CHECK FOR LACK OF PRIVILEGE
              01    13    1035  1870          BEQL    1$                              ; WHICH IS THE ONLY ACCEPTABLE ERROR
                               04    1037  1871          RET                             ; OTHERWISE GET OUT
                                    1038  1872 1$:       $SETEXV_S       ADDRES=W^EXCEPT,-       ;
                                    1038  1873                           ACMODE=#PSL$C_USER,-    ;
                                    1038  1874                           VECTOR=#0       ; SET PRIMARY FOR USER
                                    1049  1875          $SETEXV_S       ADDRES=W^CATCHALL,-     ; SET LAST CHANCE HANDLER
                                    1049  1876                           ACMODE=#PSL$C_USER,-    ; FOR USER MODE
                                    1049  1877                           VECTOR=#2       ; SPECIFY LAST CHANCE HANDLER
                                    105A  1878          $DCLEXH_S       EXITBLK         ; DECLARE USER MODE EXIT HANDLER
              3C    BB    1065  1879          PUSHR   #^M<R2,R3,R4,R5>
     54   EF95 CF    9E    1067  1880          MOVAB   W^DELBASE, R4           ; Set page protection on entire
     55   15C8'CF    9E    106C  1881          MOVAB   W^DELEND, R5            ; DELTA image to user writeable.
                0350    30    1071  1882          BSBW    SETWRT
              3C    BA    1074  1883          POPR    #^M<R2,R3,R4,R5>
                                    1076  1884          $TRNLOG_S       LOGNAM=DBGINPUT,-       ; FIRST DEFAULT INPUT
                                    1076  1885                          RSLLEN=TRNINPUT,-
                                    1076  1886                          RSLBUF=TRNINPUT
             50 50    E9    108F  1887          BLBC    R0,9$                           ; ON ERROR, USE TT
00000629 8F   50    D1    1092  1888          CMPL    R0,#SS$_NOTRAN
              47    13    1099  1889          BEQL    9$                              ; USE TT IF NO TRANSLATION
                                    1098  1890 3$:       $TRNLOG_S       LOGNAM=TRNINPUT,-       ; TRANSLATE ALL THE WAY
                                    1098  1891                          RSLLEN=TRNINPUT,-
                                    1098  1892                          RSLBUF=TRNINPUT
00000629 8F   50    D1    1084  1893          CMPL    R0,#SS$_NOTRAN
              DE    12    1088  1894          BNEQ    3$                              ; TRY ANOTHER LEVEL
```

```
                                                H 2
                1B   F3CF DF    91  10BD  1895          CMPB    @TRNINPUT+4,#^X1B       ; CHECK FOR PROCESS PERMANENT
                           0A    12  10C2  1896          BNEQ    5$
                F3C3 CF    04    C2  10C4  1897          SUBL    #4,TRNINPUT             ; REMOVE THE ESCAPE HEADER
                F3C2 CF    04    C0  10C9  1898          ADDL    #4,TRNINPUT+4
                                     10CE  1899 5$:       $ASSIGN_S       TRNINPUT,TTCHAN ; DO THE ASSIGN
                      15 50    E8  10DF  1900          BLBS    R0,10$                  ; TRY TT ON ERROR
                                     10E2  1901 9$:       $ASSIGN_S       TTNAMD,TTCHAN   ; ASSIGN DEVICE
                      01 50    E8  10F3  1902          BLBS    R0,10$                  ; CONTINUE IF SUCCESS
                               04  10F6  1903          RET                             ; ELSE EXIT WITH ERROR CODE IN R0
                54   FEAD CF    9E  10F7  1904 10$:     MOVAB   SALUTE,R4               ; SET ADDRESS OF SALUTATION
                         F7E6     30  10FC  1905          BSBW    OUTZSTRING              ; OUTPUT IT
                03 18 AC   10    E1  10FF  1906          BBC     #CLI$V_DBGEXCP,24(AP),15$       ; BR IF LATER INVOCATION
                         FEFD     31  1104  1907          BRW     NOBRK                   ; VIA $DEBUG COMMAND
                   0C'AF   6C    FA  1107  1908 15$:     CALLG   (AP),B^20$              ; CREATE TOP CALL FRAME
                               04  110B  1909          RET                             ;
                           0000    110C  1910 20$:      .WORD   0                       ; NULL ENTRY MASK
                   04 AC   04    C0  110E  1911          ADDL    #4,4(AP)                ; ADVANCE STARTING ADDRESS POINTER
                           7E    DC  1112  1912          MOVPSL  -(SP)                   ; SAVE PSL
             7E    04 BC   02    C1  1114  1913          ADDL3   #2,@4(AP),-(SP)         ; FETCH CURRENT STARTING ADDRESS
                7E   046C 8F    3C  1119  1914          MOVZWL  #$$$_DEBUG,-(SP)        ; SET EXCEPTION CODE
                           03    DD  111E  1915          PUSHL   #3                      ; SIGNAL ARG COUNT
                      50   5E    D0  1120  1916          MOVL    SP,R0                   ; SAVE POINTER
                      7E   50    7D  1123  1917          MOVQ    R0,-(SP)                ; SAVE PHONY R0,R1
                           00    DD  1126  1918          PUSHL   #0                      ; DEPTH
                           5D    DD  1128  1919          PUSHL   FP                      ; FP
                           04    DD  112A  1920          PUSHL   #4                      ; ARG COUNT
                           5E    DD  112C  1921          PUSHL   SP                      ; POINTER TO MECH
                           50    DD  112E  1922          PUSHL   R0                      ; POINTER TO SIGNAL
                11CD'CF    02    FB  1130  1923          CALLS   #2,W^EXCEPT             ; SIGNAL PHONY EXCEPTION
                      5E   0C    C0  1135  1924          ADDL    #12,SP                  ; CLEAN BACK TO R0,R1
                      50   8E    7D  1138  1925          MOVQ    (SP)+,R0                ; RESTORE R0,R1
                      5E   08    C0  113B  1926          ADDL    #8,SP                   ; CLEAN BACK TO PC,PSL
                               02  113E  1927          REI                             ; RETURN TO TARGET PROGRAM
                                   113F  1928
                                   113F  1929
                                   113F  1930          .ENABLE         LOCAL_BLOCK
                                   113F  1931
                           0000    113F  1932 SETKEXC:          .WORD   0               ; ENTRY MASK FOR CMKRNL PRIVILEGE
                   12FA'CF    9F  1141  1933          PUSHAB  W^CLREXV_KERNEL         ; SET TO USE KERNEL RUNDOWN HANDLER
                1361'CF    01    FB  1145  1934          CALLS   #1,W^SETRUNDWN          ; SET UP APPROPRIATE RUNDOWN HANDLER
                      7F   50    E9  114A  1935          BLBC    R0,20$                  ; BRANCH IF CAN'T SET UP HANDLER
                                   114D  1936          $SETEXV_S       ADDRES=B^EXCEPT,-       ;
                                   114D  1937                          PRVHND=KCOND_PRIMARY,-  ;
                                   114D  1938                          ACMODE=#PSL$C_KERNEL,-  ; SET KERNEL
                                   114D  1939                          VECTOR=#0               ; PRIMARY VECTOR
                                   115F  1940          $SETEXV_S       ADDRES=W^CATCHALL,-     ;
                                   115F  1941                          PRVHND=KCOND_LASTCHANC,-        ;
                                   115F  1942                          ACMODE=#PSL$C_KERNEL,-  ; SET KERNEL MODE LAST CHANCE HANDLE
                                   115F  1943                          VECTOR=#2               ; SPECIFY LAST CHANCE VECTOR
                      0E   11      1172  1944          BRB     10$                     ; SKIP ALTERNATE ENTRY MASK
                                   1174  1945 ;-------------------------------------------------------------------------------
                           0000    1174  1946 SETEEXC:          .WORD   0               ; ENTRY MASK FOR CMEXEC PRIVILEGE
                   131C'CF    9F  1176  1947          PUSHAB  W^CLREXV_EXEC           ; SET TO USE EXEC RUNDOWN HANDLER
                1361'CF    01    FB  117A  1948          CALLS   #1,W^SETRUNDWN          ; SET UP APPROPRIATE RUNDOWN HANDLER
                      4A   50    E9  117F  1949          BLBC    R0,20$                  ; BRANCH IF CAN'T SET UP HANDLER
                                   1182  1950 10$:       $SETEXV_S       ADDRES=B^EXCEPT,-       ;
                                   1182  1951                          PRVHND=ECOND_PRIMARY,-  ;
```

```
                                                1  2
                     1182  1952                          ACMODE=#PSL$C_EXEC,-    ; SET EXEC MODE EXCEPTION HANDLER
                     1182  1953                          VECTOR=#0               ; PRIMARY VECTOR
                     1194  1954          $SETEXV_S       ADDRES=W^CATCHALL,-     ;
                     1194  1955                          PRVHND=ECOND_LASTCHANC,-
                     1194  1956                          ACMODE=#PSL$C_EXEC,-    ; SET EXEC MODE LAST CHANCE HANDLER
                     1194  1957                          VECTOR=#2       ; SPECIFY LAST CHANCE VECTOR
                     11A7  1958 ;-------------------------------------------------------------------------
                     11A7  1959          $SETEXV_S       ADDRES=B^EXCEPT,-       ;
                     11A7  1960                          PRVHND=SCOND_PRIMARY,-  ;
                     11A7  1961                          ACMODE=#PSL$C_SUPER,-   ; SET SUPERVISOR MODE EXCEPTION HAND
                     11A7  1962                          VECTOR=#0               ; PRIMARY VECTOR
                     11B9  1963          $SETEXV_S       ADDRES=W^CATCHALL,-     ;
                     11B9  1964                          PRVHND=SCOND_LASTCHANC,-
                     11B9  1965                          ACMODE=#PSL$C_SUPER,-   ; SET SUPERVISOR LAST CHANCE HANDLER
                     11B9  1966                          VECTOR=#2       ; SPECIFY LAST CHANCE VECTOR
                04   11CC  1967 20$:     RET                             ;
                     11CD  1968
                     11CD  1969          .DISABLE        LOCAL_BLOCK
                     11CD  1970
               0000  11CD  1971 EXCEPT:  .WORD   0                       ; EXCEPTION HANDLER ENTRY MASK
                     11CF  1972          $SETEXV_S       ADDRES=B^EXCEPT,-       ;
                     11CF  1973                          ACMODE=#PSL$C_USER,-    ;
                     11CF  1974                          VECTOR=#0               ; RE-ESTABLISH USER PRIMARY VECTOR
      50  04 AC  04  C1  11DF  1975          ADDL3   #4,4(AP),R0             ; GET POINTER TO SIGNAL
          51      DC  11E4  1976          MOVPSL  R1                      ; GET CURRENT PSL
   51  51  02     18  EF  11E6  1977          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R1,R1       ;
    43 F2E4 CF    51  E2  11EB  1978          BBSS    R1,DBGACTIVE,40$        ; BR IF ALREADY ACTIVE
   60   00000464 8F  D1  11F1  1979          CMPL    #SS$_TBIT,(R0)          ; IS IT TBIT?
                 03  12  11F8  1980          BNEQ    10$                     ; NO,
              FC4B  31  11FA  1981 5$:       BRW     XDELTBIT                ; YES, A TBIT
   60   00000414 8F  D1  11FD  1982 10$:      CMPL    #SS$_BREAK,(R0)         ; IS IT BREAKPOINT?
                 03  12  1204  1983          BNEQ    20$                     ; NO,
              FB97  31  1206  1984 15$:      BRW     XDELBPT                 ; YES, A BREAKPOINT
                    1209  1985 20$:                                         ; SOME OTHER EXCEPTION
   60   00000928 8F  D1  1209  1986          CMPL    #SS$_UNWINDING,(R0)     ; IS IT UNWINDING
                 2B  13  1210  1987          BEQL    60$                     ; YES
   80   0000042C 8F  D1  1212  1988          CMPL    #SS$_COMPAT,(R0)+       ; IS IT COMPATIBILITY MODE EXCEPT?
                 0A  12  1219  1989          BNEQ    30$                     ; NO
         60      01  D1  121B  1990          CMPL    #1,(R0)                 ; IS IT COMPATIBILITY BPT?
                 E6  13  121E  1991          BEQL    15$                     ; YES
         60      07  D1  1220  1992          CMPL    #7,(R0)                 ; IS IT COMPATIBILITY TBIT?
                 D5  13  1223  1993          BEQL    5$                      ; YES
   70   0000046C 8F  D1  1225  1994 30$:      CMPL    #SS$_DEBUG,-(R0)        ; IS IT DEBUG EXCEPTION?
                 06  12  122C  1995          BNEQ    40$                     ; NO,
              FABA  30  122E  1996          BSBW    SAVE                    ; SAVE EVERYTHING
              FC21  31  1231  1997          BRW     XDELDBG                 ; AND TREAT AS FUNNY BPT
                    1234  1998 40$:                                         ; UNEXPECTED EXCEPTION
   00 F29B CF    51  E5  1234  1999          BBCC    R1,DBGACTIVE,50$        ; CLEAR DEBUG ACTIVE
                 50  D4  123A  2000 50$:      CLRL    R0                      ; RETURN FALSE FOR RESIGNAL
                 04  123C  2001              RET
         50      01  D0  123D  2002 60$:      MOVL    #1,R0                   ; IGNORE AND RESIGNAL
                 04  1240  2003              RET
```

```
                                1241  2005          .SBTTL  HANDLER FOR DEBUG EXCEPTIONS
                                1241  2006
                                1241  2007 DBGEXCEP:
                          0000  1241  2008          .WORD   0                       ;
                 51   08 AC   04   C1  1243  2009          ADDL3   #4,8(AP),R1        ; POINT TO EXCEPTION FP
                      50   5D   D0  1248  2010          MOVL    FP,R0               ; INIT LINK FOR CALL FRAMES
                      61   0C A0   D1  124B  2011 10$:     CMPL    12(R0),(R1)         ; IS THIS THE LAST ONE?
                           0B   13  124F  2012          BEQL    20$                 ; YES
                    10 A0   62'AF   9E  1251  2013          MOVAB   B^30$,16(R0)        ; SET FOR RETURN
                      50   0C A0   D0  1256  2014          MOVL    12(R0),R0           ;
                           EF   11  125A  2015          BRB     10$                 ; CONTINUE
                 10 A0   FA88 CF   9E  125C  2016 20$:     MOVAB   XDELACV,16(R0)      ; SET RETURN FOR ERROR
                           04   12  1262  2017 30$:     RET                         ;
                                1263  2018
                                1263  2019 CATCHALL:                               ; CATCHALL EXCEPTION HANDLER
                          0000  1263  2020          .WORD   0                       ; ENTRY MASK
                             51   DC  1265  2021          MOVPSL  R1                  ; GET CURMOD
              51   51   02   18   EF  1267  2022          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R1,R1       ; ISOLATE CURRENT MODE
                 03 F263 CF   51   E3  126C  2023          BBCS    R1,DBGACTIVE,10$    ; MUST NOT BE DEBUGGER EXCEPTION
                             50   D4  1272  2024          CLRL    R0                  ; RESIGNAL
                             04   12  1274  2025          RET                         ;
                          FA73   30   1275  2026 10$:     BSBW    SAVE                ; SAVE EVERYTHING
                 50   04 AC   04   C1  1278  2027          ADDL3   #4,4(AP),R0         ; POINT TO EXCEPTION CODE
                             53   60   D0  127D  2028          MOVL    (R0),R3             ; GET IT
                          F70E   30   1280  2029          BSBW    CRLF                ; OUTPUT CR/LF
                          F642   30   1283  2030          BSBW    OUTLONG             ; OUTPUT EXCEPTION CODE
                       54   90'AF   9E  1286  2031          MOVAB   B^EXCMSG,R4         ; OUTPUT MESSAGE
                          F658   30   128A  2032          BSBW    OUTZSTRING          ; TEXT FOR EXCEPTION
                          FBC5   31   128D  2033          BRW     XDELDBG             ; AND DISPLAY INSTRUCTION
00 20 4E 4F 49 54 50 45 43 58 45 20   1290  2034 EXCMSG: .ASCIZ   / EXCEPTION /       ;
                                129C  2035
                                129C  2036 EXIHANDLE:                               ; EXIT HANDLER
                          0000  129C  2037          .WORD   0                       ; ENTRY MASK
                    F231 CF   0F   93  129E  2038          BITB    #15,DBGACTIVE       ; TEST FOR DEBUG ACTIVE IN ANY MODE
                             01   13  12A3  2039          BEQL    10$                 ; NO, REPORT EXIT
                             04   12  12A5  2040          RET                         ; RETURN
                                12A6  2041 10$:                                     ; PROGRAM EXIT
                             7E   DC  12A6  2042          MOVPSL  -(SP)               ; BUILD EXCEPTION FRAME
                          10 AD   DD  12A8  2043          PUSHL   16(FP)              ;
                          04 BC   DD  12AB  2044          PUSHL   @4(AP)              ; EXIT CODE FOR EXCEPTION CODE
                             03   DD  12AE  2045          PUSHL   #3                  ; ARG COUNT
                             03   BB  12B0  2046          PUSHR   #^M<R0,R1>          ;
                       7E    5C   7D  12B2  2047          MOVQ    AP,-(SP)            ;
                             04   DD  12B5  2048          PUSHL   #4                  ; MECHANISM COUNT
                             5E   DD  12B7  2049          PUSHL   SP                  ; POINTER TO MECHANISM
                          18 AE   DF  12B9  2050          PUSHAL  24(SP)              ; POINTER TO SIGNAL
                             02   DD  12BC  2051          PUSHL   #2                  ;
                       5C    5E   D0  12BE  2052          MOVL    SP,AP               ; SET AP FOR EXCEPTION
                          FA27   30   12C1  2053          BSBW    SAVE                ; SAVE EVERYTHING
                    54    F1'AF   9E  12C4  2054          MOVAB   B^EXIMSG,R4         ; DISPLAY EXIT MESSAGE
                          F61A   30   12C8  2055          BSBW    OUTZSTRING          ; OUTPUT TEXT
                    53    48 AB   D0  12CB  2056          MOVL    SAVAP-B(R11),R3     ; GET POINTER TO EXCEPTION ARGLIST
                    53    04 A3   D0  12CF  2057          MOVL    4(R3),R3            ; GET EXIT CODE ADDRESS
                          F5F2   30   12D3  2058          BSBW    OUTLONG             ; DISPLAY IT
                                12D6  2059          $DCLEXH_S       EXITBLK         ; RE-ESTABLISH EXIT HANDLER
                             51   DC  12E1  2060          MOVPSL  R1                  ; GET CURRENT PSL
              51   51   02   18   EF  12E3  2061          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R1,R1 ; GET CURRENT MODE
```

```
                                                 K 2
                     00 F1E7 CF    51    E2  12E8  2062          BBSS    R1,DBGACTIVE,20$         ; SET DELTA ACTIVE FOR MODE
                               FB64    31  12EE  2063 20$:       BRW     XDELDBG                  ;
                                           12F1  2064
                00 20 54 49 58 45 20 0A 0D 12F1  2065 EXIMSG:    .ASCIZ  <CR><LF>/ EXIT /         ;
                                           12FA  2066
                                           12FA  2067
                                           12FA  2068          .ENABLE         LOCAL_BLOCK
                                           12FA  2069 ;
                                           12FA  2070 ; RESET INNER MODE EXCEPTION VECTORS. THIS CODE IS CALLED AS A
                                           12FA  2071 ; PRIVILEGED IMAGE RUNDOWN HANDLER TO ENSURE ITS EXECUTION.
                                           12FA  2072 ;
                                           12FA  2073 ;  THIS ENTRY POINT IS USED IF THE PROCESS HAS CMKRNL PRIVILEGE
                                           12FA  2074
                                           12FA  2075 CLREXV_KERNEL:                            ; CLEAR EXCEPTION VECTORS
                                           12FA  2076 ;
                                           12FA  2077 ;  RESET PRIMARY AND LAST CHANCE EXCEPTION VECTORS FOR KERNEL MODE
                                           12FA  2078 ;
                                           12FA  2079          $SETEXV_S       ADDRES=@KCOND_PRIMARY,- ;
                                           12FA  2080                          ACMODE=#PSL$C_KERNEL,-  ;
                                           12FA  2081                          VECTOR=#0               ; PRIMARY VECTOR
                                           130B  2082          $SETEXV_S       ADDRES=@KCOND_LASTCHANC,-       ;
                                           130B  2083                          ACMODE=#PSL$C_KERNEL,-  ;
                                           130B  2084                          VECTOR=#2               ; LAST CHANCE VECTOR
                                           131C  2085
                                           131C  2086 ;  THIS ENTRY POINT IS USED IF THE PROCESS HAS CMEXEC PRIVILEGE
                                           131C  2087
                                           131C  2088 CLREXV_EXEC:                              ; CLEAR EXCEPTION VECTORS
                                           131C  2089 ;
                                           131C  2090 ;  RESET PRIMARY AND LAST CHANCE EXCEPTION VECTORS FOR EXECUTIVE MODE
                                           131C  2091 ;
                                           131C  2092          $SETEXV_S       ADDRES=@ECOND_PRIMARY,- ;
                                           131C  2093                          ACMODE=#PSL$C_EXEC,-    ;
                                           131C  2094                          VECTOR=#0               ; PRIMARY VECTOR
                                           132D  2095          $SETEXV_S       ADDRES=@ECOND_LASTCHANC,-       ;
                                           132D  2096                          ACMODE=#PSL$C_EXEC,-    ;
                                           132D  2097                          VECTOR=#2               ; LAST CHANCE VECTOR
                                           133E  2098 ;
                                           133E  2099 ;  RESET PRIMARY AND LAST CHANCE EXCEPTION VECTORS FOR SUPERVISOR MODE
                                           133E  2100 ;
                                           133E  2101          $SETEXV_S       ADDRES=@SCOND_PRIMARY,- ;
                                           133E  2102                          ACMODE=#PSL$C_SUPER,-   ;
                                           133E  2103                          VECTOR=#0               ; PRIMARY VECTOR
                                           134F  2104          $SETEXV_S       ADDRES=@SCOND_LASTCHANC,-       ;
                                           134F  2105                          ACMODE=#PSL$C_SUPER,-   ;
                                           134F  2106                          VECTOR=#2               ; LAST CHANCE VECTOR
                                05    1360  2107          RSB                                 ;
                                           1361  2108
                                           1361  2109          .DISABLE        LOCAL_BLOCK
```

```
                                1361  2111          .SBTTL  SETRUNDWN - SET UP RUNDOWN HANDLER
                                1361  2112
                                1361  2113 ;++
                                1361  2114 ;
                                1361  2115 ; FUNCTIONAL DESCRIPTION:
                                1361  2116 ;
                                1361  2117 ;       This routine inserts the specified routine entry point into
                                1361  2118 ;       the process' rundown vector. CMKRNL privilege of running in
                                1361  2119 ;       kernel or exec mode is required.
                                1361  2120 ;
                                1361  2121 ; CALLING SEQUENCE:
                                1361  2122 ;       CALLx SETRUNDWN
                                1361  2123 ;
                                1361  2124 ; INPUT PARAMETERS:
                                1361  2125 ;       4(AP): address of rundown handler routine
                                1361  2126 ;
                                1361  2127 ; IMPLICIT INPUTS:
                                1361  2128 ;       NONE
                                1361  2129 ;
                                1361  2130 ; OUTPUT PARAMETERS:
                                1361  2131 ;       NONE
                                1361  2132 ;
                                1361  2133 ; IMPLICIT OUTPUTS:
                                1361  2134 ;       NONE
                                1361  2135 ;
                                1361  2136 ; COMPLETION CODES:
                                1361  2137 ;       SS$_NORMAL if successfully completed
                                1361  2138 ;       SS$_NOPRIV if not privileged
                                1361  2139 ;
                                1361  2140 ; SIDE EFFECTS:
                                1361  2141 ;       NONE
                                1361  2142 ;
                                1361  2143 ;--
                                1361  2144
                                1361  2145          .WEAK   CTL$GL_USRUNDWN
                                1361  2146
                                1361  2147 SETRUNDWN:
                          0000  1361  2148          .WORD   ^M<>
                       50   DC  1363  2149          MOVPSL  R0                              ; get PSL of caller
03   50   02          16   ED  1365  2150          CMPZV   #PSL$V_PRVMOD,#PSL$S_PRVMOD,R0,#PSL$C_USER
                       10   1F  136A  2151          BLSSU   5$                              ; branch if DELTA started up in inner mode
                       5C   DD  136C  2152          PUSHL   AP                              ; build $CMKRNL arg list on stack
          00001380'EF      9F  136E  2153          PUSHAB  10$                             ; address of kernel mode routine
 00000000'GF           02   FB  1374  2154          CALLS   #2,G^SYS$CMKRNL                 ; call actual routine in kernel mode
                       04   137B  2155          RET
                            137C  2156
          50   01       D0  137C  2157 5$:       MOVL    #SS$_NORMAL,R0                  ; DELTA started up in an inner mode
                       04   137F  2158          RET                                     ; do not set up a rundown handler
                            1380  2159
                            1380  2160 ;
                            1380  2161 ; The following code executes in kernel mode and actually inserts the
                            1380  2162 ; entry point into the rundown vector.
                            1380  2163 ;
                            1380  2164
                          0000  1380  2165 10$:      .WORD   ^M<>                            ; save no registers
                            1382  2166          SETIPL  #IPL$_ASTDEL                    ; lock out AST's while we modify the vector
51    00000000'GF       9E  1385  2167          MOVAB   G^CTL$GL_USRUNDWN, R1           ; Get rundown vector pointer address.
```

```
                 2E  13  138C  2168          BEQL    19$                     ; Branch to NOP routine if no present.
     51    61    04  C3  138E  2169          SUBL3   #4,(R1),R1              ; Get address of rundown vector.
50   00002034   8F  D0  1392  2170          MOVL    #SS$_VECFULL,R0
000000F9   8F    61  D1  1399  2171          CMPL    (R1),#<256-7>           ; check if another vector will fit
                 1D  1E  139D  2172          BGEQU   20$                     ; branch if not
     50    51    61  C1  13A2  2173          ADDL3   (R1),R1,R0              ; point to free vector
     80   9F16   8F  B0  13A6  2174          MOVW    #^X9F16,(R0)+           ; insert JSB @#...
       80    04  AC  D0  13AB  2175          MOVL    4(AP),(R0)+             ; insert routine address
             80  05  90  13AF  2176          MOVB    #^X05,(R0)+             ; insert final RSB
             61  06  C0  13B2  2177          ADDL    #6,(R1)                 ; update end pointer
00000004'GF      06  A0  13B5  2178          ADDW    #6,G^IAC$AW_VECSET+4    ; adjust image activation end pointer
             50  01  D0  13BC  2179 19$:     MOVL    #SS$_NORMAL,R0          ; indicate success
                     13BF  2180 20$:     SETIPL  #0                      ; enable AST's again
                 04  13C2  2181          RET
```

```
                        13C3  2183          .SBTTL  SETWRT - SET PAGES WRITABLE
                        13C3  2184 ;
                        13C3  2185 ; MAKE SPECIFIED RANGE OF PAGES WRITABLE
                        13C3  2186 ;
                        13C3  2187 ;       R4 = STARTING ADDRESS
                        13C3  2188 ;       R5 = ENDING ADDRESS
                        13C3  2189 ;
                        13C3  2190 ; R0-R2 DESTROYED
                        13C3  2191 ;
                        13C3  2192
                        13C3  2193 SETWRT:                                              ;
    7E   54   7D        13C3  2194          MOVQ    R4,-(SP)                            ; PUSH RANGE ONTO STACK
    52   7E   DE        13C6  2195          MOVAL   -(SP),R2                            ; ADDRESS FOR RETURN OF PROT
                        13C9  2196          $CMKRNL_S B^SETPRTK,(R2)                    ; TRY IN KERNEL MODE FIRST
      04 50   E8        13D5  2197          BLBS    R0,10$                              ; CONTINUE IF NO ERROR
E2'AF    62   FA        13D8  2198          CALLG   (R2),B^SETPRTK                      ; IF NOT ENOUGH PRIV, TRY USER MODE
        04   BA        13DC  2199 10$:      POPR    #^M<R2>                             ; RESTORE PROTECTION VALUE
    5E   08   C0        13DE  2200          ADDL    #8,SP                               ; CLEAN STACK
              05        13E1  2201          RSB                                         ; RETURN
                        13E2  2202
              0000      13E2  2203 SETPRTK: .WORD   0                                   ;
                        13E4  2204          $SETPRT_S       INADR=4(AP),-               ;
                        13E4  2205                          PROT=#PRT$C_UW,-            ; WRITABLE BY ALL
                        13E4  2206                          ACMODE=#0,-                 ;
                        13E4  2207                          PRVPRT=(AP)                 ; ADDRESS AT WHICH TO RETURN PROT
    50   01   D0        13F6  2208          MOVL    #1,R0                               ; ALWAYS SUCCESS
              04        13F9  2209          RET                                         ;
                        13FA  2210
                        13FA  2211 REPROT:                                              ; RESTORE PROTECTION
              05        13FA  2212          RSB                                         ;
```

```
                        13FB  2214          .SBTTL  FETCHP - FETCH DATA FROM ANOTHER PROCESS
                        13FB  2215
                        13FB  2216 FETCHP: CASE    CURTYPE-B(R11),TYPE=B,<-
                        13FB  2217                 10$,-                   ; 0 => BYTE
                        13FB  2218                 20$,-                   ; 1 => WORD
                        13FB  2219                 30$>                    ; 2 => LONG
                     05 1406  2220          RSB                             ; UNKNOWN
14A3'CF                 9F 1407  2221 10$:     PUSHAB  W^FPBYTE                ; SET FOR BYTE FETCH
                     0A 11 140B  2222          BRB     40$
1505'CF                 9F 140D  2223 20$:     PUSHAB  W^FPWORD                ; SET FOR WORD FETCH
                     04 11 1411  2224          BRB     40$
1567'CF                 9F 1413  2225 30$:     PUSHAB  W^FPLONG                ; SET FOR LONGWORD FETCH
                  F0 AB DD 1417  2226 40$:     PUSHL   PID-B(R11)              ; PID OF TARGET PROCESS
                  04 AB 9F 141A  2227          PUSHAB  QUAN-B(R11)             ; SET ADDRESS TO RETURN VALUE
                     6B DD 141D  2228          PUSHL   CURDOT-B(R11)           ; AND ADDRESS OF VALUE
                     04 DD 141F  2229          PUSHL   #4                      ; ARGUMENT COUNT
                  50 5E DO 1421  2230          MOVL    SP,R0                   ; SAVE POINTER TO ARG LIST
                           1424  2231          $CMKRNL_S       W^QGET,(R0)     ; Q AST FOR DATA FETCH
                  07 50 E9 1431  2232          BLBC    R0,50$                  ; BR IF FAILED
                           1434  2233          $HIBER_S                        ; WAIT FOR DATA TO RETURN
                  5E 14 C0 143B  2234 50$:     ADDL    #20,SP                  ; CLEAN STACK
                        143E  2235          RSB                             ; AND RETURN DATA
```

```
                                143F  2237          .SBTTL  QGET - QUEUE AST TO GET DATA FROM ANOTHER PROCESS
                                143F  2238 ;
                                143F  2239 ;       INPUTS: 04(AP) - LOCATION OF DATA
                                143F  2240 ;               08(AP) - RETURN LOCATION
                                143F  2241 ;               12(AP) - PID OF TARGET PROCESS
                                143F  2242 ;               16(AP) - CODE SEGMENT POINTER
                                143F  2243 ;
                                143F  2244          .WEAK   EXE$ALLOCBUF            ; MAKE FOLLOWING CODE OPTIONAL
                                143F  2245          .WEAK   EXE$DEANONPAGED
                                143F  2246          .WEAK   SCH$WAKE
                                143F  2247          .WEAK   SCH$QAST
                                143F  2248          .WEAK   SCH$GL_MAXPIX
                                143F  2249
                    00000010    143F  2250          FP_ORIGPID=ACB$L_AST
                    00000014    143F  2251          FP_ADDR=ACB$L_ASTPRM
                    00000014    143F  2252          FP_VALUE=ACB$L_ASTPRM
                    0000001C    143F  2253          FP_RETLOC=ACB$L_KAST+4
                                143F  2254
                         003C   143F  2255  QGET:    .WORD   ^M<R2,R3,R4,R5>         ; ENTRY MASK
          50    08E8 8F    3C   1441  2256          MOVZWL  #SS$_NONEXPR,R0         ; ASSUME BAD PIX
    51    00000000'GF    9E   1446  2257          MOVAB   G^EXE$ALLOCBUF,R1       ; WERE WE LINKED WITH SYS.STB SYMBOLS?
                   53    13   144D  2258          BEQL    10$                     ; IF NOT, RETURN WITH ERROR
00000000'9F    0C AC    B1   144F  2259          CMPW    12(AP),@#SCH$GL_MAXPIX  ; CHECK PIX FOR LEGAL PROCESS
                   49    1A   1457  2260          BGTRU   10$                     ; BR IF NOT
```

```
                51    10 BC    3C   1459  2262          MOVZWL  @16(AP),R1                     ; GET SIZE OF CODE SEGMENT
              51    00C4 C1    9E   145D  2263          MOVAB   IRP$C_LENGTH(R1),R1           ; ADD SIZE OF PACKET DATA
               00000000'9F     16   1462  2264          JSB     @#EXE$ALLOCBUF                ; ALLOCATE BUFFER TO CONTAIN CODE
                     37 50     E9   1468  2265          BLBC    R0,10$                        ; BRANCH IF NONE
                  55    52     D0   146B  2266          MOVL    R2,R5                         ; SAVE ADDRESS OF PACKET
           10 A5    60 A4     D0   146E  2267          MOVL    PCB$L_PID(R4),FP_ORIGPID(R5)      ; SET PID FOR RETURN
           0B A5    80 BF     90   1473  2268          MOVB    #^X80,ACB$B_RMOD(R5)          ; SET FOR SPECIAL KERNEL AST
           18 A5    20 A5     9E   1478  2269          MOVAB   ACB$L_KAST+8(R5),ACB$L_KAST(R5) ; SET ADDRESS FOR AST
           14 A5    04 AC     D0   147D  2270          MOVL    4(AP),FP_ADDR(R5)             ; SET ADDRESS FOR FETCH
           1C A5    08 AC     D0   1482  2271          MOVL    8(AP),FP_RETLOC(R5)           ; AND ADDRESS OF RETURN LOCATION
              50    10 AC     D0   1487  2272          MOVL    16(AP),R0                     ; GET ADDRESS OF CODE SEGMENT
           0C A5    0C AC     D0   148B  2273          MOVL    12(AP),ACB$L_PID(R5)          ; SET TARGET PID
                     3F     BB   1490  2274          PUSHR   #^M<R0,R1,R2,R3,R4,R5>        ; SAVE REGS FOR MOVC
        20 A5    60    80     28   1492  2275          MOVC3   (R0)+,(R0),ACB$L_KAST+8(R5); COPY CODE SEGMENT TO BUFFER
                     3F     BA   1497  2276          POPR    #^M<R0,R1,R2,R3,R4,R5>        ; RESTORE REGISTERS
                  52    04     9A   1499  2277          MOVZBL  #PRI$_TICOM,R2                ; SET PRIORITY INCREMENT CLASS
               00000000'9F     16   149C  2278          JSB     @#SCH$QAST                    ; QUEUE AST FOR TARGET
                           04   14A2  2279 10$:         RET                                   ; RETURN TO ORIGINAL MODE
                                14A3  2280
                                14A3  2281          .SBTTL  FPBYTE - FETCH BYTE FROM PROCESS
                          0049' 14A3  2282 FPBYTE:    .WORD   90$-.-2                       ; SIZE OF CODE SEGMENT
                                14A5  2283          IFNORD  #1,@FP_ADDR(R5),10$           ; BRANCH IF NOT READABLE
           14 A5    14 B5     90   14AC  2284          MOVB    @FP_ADDR(R5),FP_VALUE(R5)     ; GET VALUE
           0C A5    10 A5     D0   14B1  2285 10$:        MOVL    FP_ORIGPID(R5),ACB$L_PID(R5) ; SET PID FOR RETURN AST
           0B A5    80 8F     90   14B6  2286          MOVB    #^X80,ACB$B_RMOD(R5)          ; SET FOR KAST AGAIN
           18 A5    C9'AF     9E   14BB  2287          MOVAB   B^20$,ACB$L_KAST(R5)          ; SET NEW AST ADDRESS
                  52    04     9A   14C0  2288          MOVZBL  #PRI$_TICOM,R2                ; SET PRIORITY INCREMENT CLASS
               00000000'9F     17   14C3  2289          JMP     @#SCH$QAST                    ; QUEUE RETURN AST
                                14C9  2290 20$:         IFNOWRT #1,@FP_RETLOC(R5),30$         ; IF NOT WRITABLE THEN SKIP IT
        1C B5    14 A5     90   14D0  2291          MOVB    FP_VALUE(R5),@FP_RETLOC(R5)       ; RETURN VALUE
              51    0C A5     D0   14D5  2292 30$:        MOVL    ACB$L_PID(R5),R1              ; GET PID FOR WAKE
                                14D9  2293          SETIPL  #IPL$_SYNCH                   ; RAISE TO SYNCH
               00000000'9F     16   14DC  2294          JSB     @#SCH$WAKE                    ; WAKE PROCESS
                                14E2  2295          SETIPL  #IPL$_ASTDEL                  ; LOWER IPL
                  50    55     D0   14E5  2296          MOVL    R5,R0                         ; SET ADDRESS FOR RELEASE
               00000000'9F     17   14E8  2297          JMP     @#EXE$DEANONPAGED             ; FREE BLOCK AND EXIT
                                14EE  2298 90$:                                           ; END OF CODE SEGMENT
                                14EE  2299
```

```
                        14EE  2301                .SBTTL  DPBYTE - DEPOSIT BYTE TO PROCESS
                0015'   14EE  2302 DPBYTE:        .WORD   90$-.-2                 ; SIZE OF CODE SEGMENT
                        14F0  2303 20$:          IFNOWRT #1,@FP_RETLOC(R5),30$   ; IF NOT WRITABLE THEN SKIP IT
      1C B5    14 A5   90  14F7  2304                MOVB    FP_VALUE(R5),@FP_RETLOC(R5)     ; RETURN VALUE
          50     55    D0  14FC  2305 30$:          MOVL    R5,R0                   ; SET ADDRESS FOR RELEASE
    00000000'9F         17  14FF  2306                JMP     @#EXE$DEANONPAGED       ; FREE BLOCK AND EXIT
                        1505  2307 90$:                                            ; END OF CODE SEGMENT
                        1505  2308
```

```
                        1505  2310                .SBTTL  FPWORD - FETCH WORD FROM PROCESS
                0049'   1505  2311 FPWORD:        .WORD   90$-.-2                  ; SIZE OF CODE SEGMENT
                        1507  2312                IFNORD  #2,@FP_ADDR(R5),10$      ; BRANCH IF NOT READABLE
14 A5   14 B5   B0      150E  2313                MOVW    @FP_ADDR(R5),FP_VALUE(R5) ; GET VALUE
0C A5   10 A5   D0      1513  2314 10$:          MOVL    FP_ORIGPID(R5),ACB$L_PID(R5) ; SET PID FOR RETURN AST
08 A5   80 8F   90      1518  2315                MOVB    #^X80,ACB$B_RMOD(R5)     ; SET FOR KAST AGAIN
18 A5   2B'AF   9E      151D  2316                MOVAB   B^20$,ACB$L_KAST(R5)     ; SET FOR NEW AST ADDRESS
     52   04   9A      1522  2317                MOVZBL  #PRI$_TICOM,R2           ; SET PRIORITY INCREMENT CLASS
00000000'9F   17        1525  2318                JMP     @#SCH$QAST               ; QUEUE RETURN AST
                        152B  2319 20$:          IFNOWRT #2,@FP_RETLOC(R5),30$    ; IF NOT WRITABLE THEN SKIP IT
1C B5   14 A5   B0      1532  2320                MOVW    FP_VALUE(R5),@FP_RETLOC(R5)      ; RETURN VALUE
   51   0C A5   D0      1537  2321 30$:          MOVL    ACB$L_PID(R5),R1         ; GET PID FOR WAKE
                        153B  2322                SETIPL  #IPL$_SYNCH              ; RAISE TO SYNCH
00000000'9F   16        153E  2323                JSB     @#SCH$WAKE               ; WAKE PROCESS
                        1544  2324                SETIPL  #IPL$_ASTDEL             ; LOWER IPL
     50   55   D0      1547  2325                MOVL    R5,R0                    ; SET ADDRESS FOR RELEASE
00000000'9F   17        154A  2326                JMP     @#EXE$DEANONPAGED        ; FREE BLOCK AND EXIT
                        1550  2327 90$:                                           ; END OF CODE SEGMENT
                        1550  2328
```

```
                        1550  2330                .SBTTL  DPWORD - DEPOSIT WORD TO PROCESS
                0015'   1550  2331 DPWORD:        .WORD   90$-.-2                 ; SIZE OF CODE SEGMENT
                        1552  2332 20$:           IFNOWRT #2,@FP_RETLOC(R5),30$   ; IF NOT WRITABLE THEN SKIP IT
1C B5   14 A5   B0      1559  2333                MOVW    FP_VALUE(R5),@FP_RETLOC(R5)     ; RETURN VALUE
     50    55   D0      155E  2334 30$:           MOVL    R5,R0                   ; SET ADDRESS FOR RELEASE
  00000000'9F   17      1561  2335                JMP     @#EXE$DEANONPAGED               ; FREE BLOCK AND EXIT
                        1567  2336 90$:                                           ; END OF CODE SEGMENT
                        1567  2337
```

```
                                1567  2339
                                1567  2340 FPLONG:  .SBTTL  FPLONG - FETCH LONG FROM PROCESS
                        0048'   1567  2340           .WORD   90$-.-2                 ; SIZE OF CODE SEGMENT
                                1569  2341           IFNORD  #4,@FP_ADDR(R5),10$     ; BRANCH IF NOT READABLE
14 A5    14 B5    D0            1570  2342           MOVL    @FP_ADDR(R5),FP_VALUE(R5) ; GET VALUE
0C A5    10 A5    D0            1575  2343 10$:      MOVL    FP_ORIGPID(R5),ACB$L_PID(R5) ; SET PID FOR RETURN AST
0B A5    80 8F    90            157A  2344           MOVB    #^X80,ACB$B_RMOD(R5)    ; SET FOR KAST AGAIN
18 A5    BC'AF    9E            157F  2345           MOVAB   B^20$,ACB$L_KAST(R5)    ; SET NEW KAST ADDRESS
                52   D4          1584  2346           CLRL    R2                      ; NULL PRIO INCR
          00000000'9F   17      1586  2347           JMP     @#SCH$QAST              ; QUEUE RETURN AST
                                158C  2348 20$:       IFNOWRT #4,@FP_RETLOC(R5),30$   ; IF NOT WRITABLE THEN SKIP IT
1C B5    14 A5    D0            1593  2349           MOVL    FP_VALUE(R5),@FP_RETLOC(R5)     ; RETURN VALUE
   51    0C A5    D0            1598  2350 30$:      MOVL    ACB$L_PID(R5),R1        ; GET PID FOR WAKE
                                159C  2351           SETIPL  #IPL$_SYNCH             ; RAISE TO SYNCH
          00000000'9F   16      159F  2352           JSB     @#SCH$WAKE              ; WAKE PROCESS
                                15A5  2353           SETIPL  #IPL$_ASTDEL            ; LOWER IPL
       50    55    D0            15A8  2354           MOVL    R5,R0                   ; SET ADDRESS FOR RELEASE
          00000000'9F   17      15AB  2355           JMP     @#EXE$DEANONPAGED       ; FREE BLOCK AND EXIT
                                15B1  2356 90$:                                       ; END OF CODE SEGMENT
                                15B1  2357
```

```
                     15B1  2359                .SBTTL  DPLONG - DEPOSIT LONGWORD TO PROCESS
                0015' 15B1  2360 DPLONG:       .WORD   90$-.-2                 ; SIZE OF CODE SEGMENT
                      15B3  2361 20$:          IFNOWRT #4,@FP_RETLOC(R5),30$   ; IF NOT WRITABLE THEN SKIP IT
 1C B5   14 A5   D0   15BA  2362                MOVL    FP_VALUE(R5),@FP_RETLOC(R5)     ; RETURN VALUE
      50    55   D0   15BF  2363 30$:          MOVL    R5,R0                   ; SET ADDRESS FOR RELEASE
00000000'9F     17   15C2  2364                JMP     @#EXE$DEANONPAGED       ; FREE BLOCK AND EXIT
                     15C8  2365 90$:                                          ; END OF CODE SEGMENT
                     15C8  2366 DELEND:                                       ;
                     15C8  2367                .ENDC                          ;
                     15C8     1                .END    TEST                   ; DECLARE START ADDRESS
```

```
$$T1                    = 00000000
ACB$B_RMOD              = 0000000B
ACB$L_AST               = 00000010
ACB$L_ASTPRM            = 00000014
ACB$L_KAST              = 00000018
ACB$L_PID               = 0000000C
ADD                       00000606 R      02
ASTEN                     000000EC R      02
B                         0000008C R      02
BLANK                     00000A55 R      02
BMSG                      00000D96 R      02
BRKADR                  = 0000039C R      02
BRKCOM                  = 000003E9 R      02
BRKDSP                  = 000003C9 R      02
BRKOP                   = 000003C3 R      02
BRKPOINT                  00000B95 R      02
BSLSH                   = 0000005C
CATCHALL                  00001263 R      02
CLI$V_DBGEXCP           = 00000010
CLREXV_EXEC               0000131C R      02
CLREXV_KERNEL             000012FA R      02
COLON                     00000C58 R      02
COMMA                     00000666 R      02
CONTEXT                   0000000C R      02
CONTEXTSZ               = 000000E4
CR                      = 0000000D
CRLF                      00000991 R      02
CTL$GL_USRUNDWN           ********W GX    02
CURDOT                    0000008C R      02
CURTYPE                   0000008A R      02
DBGACTIVE                 000004D4 R      02
DBGEXCEP                  00001241 R      02
DBGINPUT                  0000047B R      02
DCOM                      0000054D R      02
DELBASE                   00000000 R      02
DELEND                    000015C8 R      02
DELTA_START               00000FC1 R      02
DEPOSIT                   00000EE3 R      02
DEPPREG                   00000F72 R      02
DIV                       00000602 R      02
DOT                       00000C68 R      02
DPBYTE                    000014EE R      02
DPLONG                    000015B1 R      02
DPWORD                    00001550 R      02
DQUOTE                    0000060A R      02
DTYPE                     00000089 R      02
ECOND_LASTCHANC           000004FC R      02
ECOND_PRIMARY             000004F0 R      02
ENDEXPR                   000005DB R      02
ENDFIELD                  00000669 R      02
EQL1                      00000AC4 R      02
EQUALS                    00000ABD R      02
ERR2                      00000B00 R      02
ERR3                      00000CE8 R      02
ERR4                      00000645 R      02
ERROR                     00000556 R      02
ESCAP                     00000A7A R      02
EXCEPT                    000011CD R      02
EXCMSG                    00001290 R      02
EXCODA                    000004E4 R      02
EXE$ALLOCBUF              ********W GX    02
EXE$DEANONPAGED           ********W GX    02
EXECUTE                   00000F95 R      02
EXIHADR                   000004DC R      02
EXIHANDLE                 0000129C R      02
EXIMSG                    000012F1 R      02
EXITBLK                   000004D8 R      02
EXITCODE                  000004E8 R      02
F1                        00000064 R      02
F2                        00000068 R      02
F3                        0000006C R      02
F4                        00000070 R      02
F5                        00000074 R      02
FCTR                      00000088 R      02
FETCH                     00000684 R      02
FETCHP                    000013FB R      02
FPBYTE                    000014A3 R      02
FPLONG                    00001567 R      02
FPWORD                    00001505 R      02
FP_ADDR                 = 00000014
FP_ORIGPID              = 00000010
FP_RETLOC               = 0000001C
FP_VALUE                = 00000014
FTCHPREG                  000006BA R      02
GETBPTX                   00000EBF R      02
GETCHAR                   0000099B R      02
GETCMD                    00000E14 R      02
GLOBL                     000005C6 R      02
GO                        00000C47 R      02
HIGH                      000005CC R      02
IAC$AW_VECSET             ********   X    02
INBUF                     00000010 R      02
INFLD                     000005C2 R      02
INIBRKA                   000003A0 R      02
INITCALL                  0000100B R      02
INSBUF                    00000084 R      02
INSLEN                    00000080 R      02
INSTR                     00000A90 R      02
IOSM_EXTEND             = 00008000
IO$_READVBLK            = 00000031
IO$_WRITEVBLK           = 00000030
IPL$_ASTDEL             = 00000002
IPL$_SYNCH              = 00000008
IRPSC_LENGTH            = 000000C4
KCOND_LASTCHANC           000004F8 R      02
KCOND_PRIMARY             000004EC R      02
LBRACKET                  00000B08 R      02
LF                      = 0000000A
LIB$INS_DECODE            ********W GX    02
LINEFEED                  000006E5 R      02
LOCOUT                    000006ED R      02
LOCP                      00000A8D R      02
LOCPROMPT                 000006EA R      02
MCHK                      00000CE8 R      02
```

```
MFYFLG                  00000078 R      02
MFYFLGS                 00000C62 R      02
MODES                   00000B03 R      02
MUL                     000005FE R      02
NBRK                =   00000008
NEGATE                  00000A5E R      02
NEXTDOT                 000006C9 R      02
NEXTLOC                 000006E8 R      02
NEXTP                   0000056D R      02
NMODES              =   00000005
NOBRK                   00001004 R      02
NPRIM               =   0000002C
NSEC                =   00000007
NTERM               =   0000000A
NTMPBRK             =   00000001
OPEN                    00000614 R      02
OPER                    0000008B R      02
OPERATOR                00000A55 R      02
OPERBAS             =   00000012
OUTB                =   00000006
OUTBB                   000006E2 R      02
OUTBSLSH                00000939 R      02
OUTBUF                  00000094 R      02
OUTCHAR                 00000942 R      02
OUTCOM                  000008CB R      02
OUTCR               =   00000004
OUTDIGIT                000008C4 R      02
OUTER                   00000545 R      02
OUTINS                  00000722 R      02
OUTLONG                 000008C8 R      02
OUTPC                   00000E01 R      02
OUTPUT                  000006F6 R      02
OUTPUTA                 0000084E R      02
OUTPUT_ADDRESS          0000079D R      02
OUTR8                   0000093F R      02
OUTSPACE                0000098C R      02
OUTZBUF                 000008E1 R      02
OUTZSTRING              000008E5 R      02
OVEROPCLEN          =   00000005
OVEROPCODES             00000514 R      02
OVRADR                  000003C0 R      02
OVROPC                  0000039C R      02
PCB$L_PID           =   00000060
PID                     0000007C R      02
PR$_IPL             =   00000012
PRET                    000005D3 R      02
PREG                    00000FA3 R      02
PREXC                   00000F81 R      02
PRI$_TICOM          =   00000004
PRIMARY                 00000519 R      02
PROCED                  00000C50 R      02
PROCEED                 00000E19 R      02
PROGCTR                 00000C7B R      02
PRT$C_UW            =   00000004
PSL$C_EXEC          =   00000001
PSL$C_KERNEL        =   00000000
PSL$C_SUPER         =   00000002
PSL$C_USER          =   00000003
PSL$S_CURMOD        =   00000002
PSL$S_PRVMOD        =   00000002
PSL$V_CURMOD        =   00000018
PSL$V_PRVMOD        =   00000016
PSL$V_TBIT          =   00000004
QGET                    0000143F R      02
QUAN                    00000090 R      02
QUANT                   00000C75 R      02
QUOT                =   00000027
QUOTE                   00000ECF R      02
RDBUF               =   00000002
RDCR                =   00000000
REGCOM                  00000C95 R      02
REGISTER                00000C8D R      02
RELOC                   0000081E R      02
REPRUT                  000013FA R      02
RESET                   00000ACC R      02
RESTORE                 00000D4A R      02
RESTORR                 00000D58 R      02
RETURN                  00000648 R      02
RSET                    0000065C R      02
RUBOUT              =   0000007F
SALUTE                  00000FA8 R      02
SAV...              =   0000039C R      02
SAVAP                   000000D4 R      02
SAVE                    00000CEB R      02
SAVOCR                  000000E8 R      02
SAVPC                   000000E0 R      02
SAVPSL                  000000E4 R      02
SAVR2                   000000AC R      02
SAVRCR                  000000EA R      02
SAVREG                  000000A4 R      02
SAVRXCS                 000000EC R      02
SAVSP                   000000DC R      02
SCANP                   00000569 R      02
SCH$GL_MAXPIX           ********W  GX   02
SCH$QAST                ********W  GX   02
SCH$WAKE                ********W  GX   02
SCOND_LASTCHANC         00000500 R      02
SCOND_PRIMARY           000004F4 R      02
SECOND                  00000AD9 R      02
SEMI                    00000AE0 R      02
SETBRK                  00000E89 R      02
SETEEXC                 00001174 R      02
SETKEXC                 0000113F R      02
SETPRTK                 000013E2 R      02
SETRUNDWN               00001361 R      02
SETWRT                  000013C3 R      02
SHFT                    000005F9 R      02
SHOBRK                  00000C05 R      02
SLASH                   0000060F R      02
SLSH                =   0000002F
SPACES                  0000071E R      02
SS$_BREAK           =   00000414
SS$_COMPAT          =   0000042C
SS$_DEBUG           =   0000046C
```

```
SS$_EXQUOTA                     = 0000001C
SS$_EXQUOTAEND                  = 00002AFF
SS$_EXQUOTASTRT                 = 00002A00
SS$_INSFMEM                     = 00000124
SS$_NONEXPR                     = 000008E8
SS$_NOPRIV                      = 00000024
SS$_NORMAL                      = 00000001
SS$_NOTRAN                      = 00000629
SS$_TBIT                        = 00000464
SS$_UNWINDING                   = 00000928
SS$_VECFULL                     = 00002034
STATUS                            00000060 R      02
STEP                              00000B37 R      02
STEPOVER                          00000B44 R      02
SUPERST                           0000055D R      02
SW_PROCESS                      = 00000001
SYS$ASSIGN                        ******** GX     02
SYS$CMEXEC                        ******** GX     02
SYS$CMKRNL                        ******** GX     02
SYS$DCLEXH                        ******** GX     02
SYS$EXIT                          ******** GX     02
SYS$HIBER                         ******** GX     02
SYS$QIOW                          ******** GX     02
SYS$SETAST                        ******** GX     02
SYS$SETEXV                        ******** GX     02
SYS$SETPRT                        ******** GX     02
SYS$TRNLOG                        ******** GX     02
SYS$WAITFR                        ******** GX     02
SYS$WAKE                          ******** GX     02
TAB                               00000A6A R      02
TERM                              00000532 R      02
TERMASK                           00000504 R      02
TERMASKADR                        00000473 R      02
TERMASKLEN                      = 00000010
TEST                              00000FBF R      02
TRM$M_TM_NOEDIT                 = 00008000
TRM$_MODIFIERS                  = 00000000
TRM$_TERM                       = 00000003
TRNINPUT                          0000048C R      02
TTCHAN                            00000455 R      02
TTIOSB                            0000044D R      02
TTITMLST                          00000463 R      02
TTITMLSTLEN                     = 00000018
TTNAMD                            00000459 R      02
UNBRK                             00000E61 R      02
VALI                              00000CBA R      02
VALR                              00000CB7 R      02
VALUE                             00000C7F R      02
V_ASCII                         = 00000001
V_ATBRK                         = 00000004
V_F1                            = 00000008
V_F2                            = 00000009
V_F3                            = 0000000A
V_F4                            = 0000000B
V_F5                            = 0000000C
V_INFIELD                       = 00000002
V_INSTR                         = 0000000D
V_NEGATE                        = 00000007
V_OPEN                          = 00000000
V_PREG                          = 0000001F
V_PRMODE                        = 0000000F
V_RUB                           = 00000006
V_TBIT                          = 00000003
V_TBITOK                        = 00000005
XDELACV                           00000CE8 R      02
XDELBPT                           00000DA0 R      02
XDELDBG                           00000E55 R      02
XDELTBIT                          00000E48 R      02
XDT$START                         00000FBF RG     02
XREG                              00000CDC R      02
XREGV                             0000040D R      02
XSET                              00000CCA R      02
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
Z$DEBUG_CODE	000015C8 (5576.)	02 (2.)	PIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.06	00:00:01.27
Command processing	148	00:00:00.66	00:00:06.91
Pass 1	536	00:00:15.18	00:01:09.72
Symbol table sort	0	00:00:02.08	00:00:09.06
Pass 2	406	00:00:04.49	00:00:21.91
Symbol table output	1	00:00:00.17	00:00:00.80
Psect synopsis output	0	00:00:00.02	00:00:00.31
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1122	00:00:22.66	00:01:49.98

The working set limit was 2250 pages.
129975 bytes (254 pages) of virtual memory were used to buffer the intermediate code.
There were 100 pages of symbol table space allocated to hold 1725 non-local and 181 local symbols.
2369 source lines were read in Pass 1, producing 26 object records in Pass 2.
41 pages of virtual memory were used to define 40 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	10
-255DUA28:[SYSLIB]STARLET.MLB;2	27
TOTALS (all libraries)	37

1738 GETS were required to define 37 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:DELTA/OBJ=OBJ$:DELTA MSRC$:SWP/UPDATE=(ENH$:SWP)+MSRC$:XDELTA/UPDATE=(ENH$:XDELTA)+MSRC$:ENDP/UPDATE=(ENH$:ENDP)+EXEC
```

SSIK
LIS

DELTA
LIS

DELETE

DELEMAIN
LIS

SSITAB
LIS

DELETE
MAP

DELTA

DELTA
MAP

SSIW
LIS

SSIUW
LIS

PURGE
LIS

STRUCDEF
LIS

SSIDISP
LIS

S0DELTA
MAP

DELETE
REQ

DIFDEF
MDL

DIF

DIFF
MAP

XDSTRING
LIS

DIFFRE
REQ

DIFGETCMD
LIS

DIFHEXOCT
LIS

DATA
LIS

XDELTA
LIS